Mathématiques pour l'informatique. L1 Informatique UE-12.

Contrôle Terminal - Session 1 (correction) - Janvier 2025

EXERCICE 1. Soit $\mathscr{B} := \{0,1\}$. Une fonction booléenne $f : \mathscr{B}^n \to \mathscr{B}$ est dite *autoduale* si et seulement si elle est égale à sa fonction duale f^*, i.e.

$$\forall (x_1,\dots,x_n) \in \mathscr{B}^n \quad f(x_1,\dots,x_n) = \overline{f(\overline{x_1},\dots,\overline{x_n})}.$$

(1) (1.0) La fonction $f : \mathscr{B}^2 \to \mathscr{B}$ définie par $f(x,y) := x\,\overline{y}$ est-elle autoduale ? La fonction première projection pr_1 est-elle autoduale ?

(2) (0.5) Combien existe-t-il de fonctions booléennes de 2 variables ?

(3) (2.0) Dressez un tableau sur 3 colonnes contenant les fonctions booléennes de 2 variables sous forme normale canonique disjonctive dans la 1ère colonne, leurs expressions simplifiées (quand c'est possible) dans la 2ème colonne, et enfin leurs fonctions duales dans la 3ème colonne.

Exemple d'une ligne de ce tableau :

$f(x,y)$	simplifiée	$f^*(x,y)$
$x\,\overline{y} + \overline{x}\,y + \overline{x}\,\overline{y}$	$\overline{x} + \overline{y}$	$\overline{x}\,\overline{y}$

(4) (0.5) Combien y-a-t-il de fonctions autoduales dans ce tableau ?

(5) (1.0) On suppose que l'on dispose de la table de vérité d'une fonction booléenne f. Proposez un algorithme qui s'appuie sur les valeurs de cette table pour vérifier que f est autoduale. Illustrez votre algorithme sur la table de vérité d'une fonction autoduale à deux variables de votre choix.

(6) (1.0) En déduire le nombre de fonctions autoduales à n variables.

Solution. (1) Soit $(x,y) \in \mathscr{B}^2$. Calculons $f^*(x,y)$:

$$\begin{aligned} f^*(x,y) &= \overline{f(\overline{x},\overline{y})} && \text{(définition)} \\ &= \overline{\overline{x}.\overline{\overline{y}}} && \text{(substitution des variables)} \\ &= \overline{\overline{x}.y} && \text{(involutivité de la négation)} \\ &= \overline{\overline{x}} + \overline{y} && \text{(loi de De Morgan)} \\ &= x + \overline{y} && \text{(involutivité de la négation)} \end{aligned}$$

Le fait que les expressions de f^* et de f soient différentes ne signifie pas que ce sont des fonctions différentes, il y a d'ailleurs une infinité d'expressions différentes d'une même fonction booléenne. C'est *précisément* la raison d'être des formes normales canoniques [1] : deux formes normales canoniques différentes décrivent des fonctions différentes.

Vérifions que $f^* \neq f$. Il suffit d'exhiber un couple (x,y) où les fonctions diffèrent, prouvant que leurs graphes sont distincts. Par exemple $(0,0)$:

$$f(0,0) = 0.\overline{0} = 0.1 = 0 \quad \text{et} \quad f^*(0,0) = 0 + \overline{0} = 0 + 1 = 1.$$

Par définition $\mathrm{pr}_1(x,y) := x$, calculons $\mathrm{pr}_1^*(x,y)$:

$$\begin{aligned} \mathrm{pr}_1^*(x,y) &= \overline{\mathrm{pr}_1(\overline{x},\overline{y})} && \text{(définition)} \\ &= \overline{\overline{x}} && \text{(substitution des variables)} \\ &= x && \text{(involutivité de la négation)} \end{aligned}$$

On a montré que pour tout couple (x,y) de booléens $\mathrm{pr}_1(x,y) = \mathrm{pr}_1^*(x,y)$, les fonctions ayant même ensemble de départ, d'arrivée et graphe, elles sont donc égales prouvant que pr_1 est autoduale.

NB. Pour les mêmes raisons, la deuxième projection pr_2 est elle aussi autoduale et plus généralement pour une fonction booléenne de n variables, les n projections pr_i, $i \in [\![1,n]\!]$ sont toutes autoduales.

(2) Chacun des n termes d'un n-uplets de booléens étant indépendant des autres et pouvant prendre deux valeurs distinctes, on peut en construire 2^n. Une fonction de n variables booléennes doit attribuer une valeur binaire à chacun de ces n-uplets, il y a donc 2^{2^n} manière de le faire. On en déduit qu'il existe $2^{2^2} = 2^4 = 16$ fonctions booléennes de $n = 2$ variables.

(3) On rappelle qu'un minterme est une fonction booléenne prenant exactement une fois la valeur 1. Pour $n = 2$, les 4 mintermes sont $x\,y$, $x\,\overline{y}$, $\overline{x}\,y$ et $\overline{x}\,\overline{y}$. Pour construire les 16 FND, il suffit de générer toutes les combinaisons additives de ces 4 mintermes. On peut les décrire dans l'ordre croissant du nombre de mintermes dans la FND ou, entre autres, en utilisant les 4 chiffres de la décomposition binaire des entiers de 0 à 15 pour ajouter ou non (1/0) le minterme correspondant dans la somme (cf. table 1).

(4) Pour vérifier qu'une FND est autoduale, soit sa duale est identique à la forme simplifiée, soit sa duale est sous FND et on peut les comparer, dans tous les autres cas, il faut s'en assurer autrement puisque plusieurs expressions différentes peuvent coder une même fonction. Dans la table 1, on a calculé les formes duales à partir des formes simplifiées. Par exemple

1. en raccourci FND pour une forme disjonctive et FNC pour une forme conjonctive

	$x\,y$|$x\,\overline{y}$|$\overline{x}\,y$|$\overline{x}\,\overline{y}$	$f(x,y)$	simplifiée	$f^*(x,y)$
0	0 0 0 0	0	0	1
1	0 0 0 1	$\overline{x}\,\overline{y}$		$\overline{x}+\overline{y}$
2	0 0 1 0	$\overline{x}\,y$		$\overline{x}+y$
3	0 0 1 1	$\overline{x}\,y+\overline{x}\,\overline{y}$	$\overline{x}$	$\overline{x}$
4	0 1 0 0	$x\,\overline{y}$		$x+\overline{y}$
5	0 1 0 1	$x\,\overline{y}+\overline{x}\,\overline{y}$	$\overline{y}$	$\overline{y}$
6	0 1 1 0	$x\,\overline{y}+\overline{x}\,y$		$x\,y+\overline{x}\,\overline{y}$
7	0 1 1 1	$x\,\overline{y}+\overline{x}\,y+\overline{x}\,\overline{y}$	$\overline{x}+\overline{y}$	$\overline{x}\,\overline{y}$
8	1 0 0 0	$x\,y$		$x+y$
9	1 0 0 1	$x\,y+\overline{x}\,\overline{y}$		$x\,\overline{y}+\overline{x}\,y$
10	1 0 1 0	$x\,y+\overline{x}\,y$	y	y
11	1 0 1 1	$x\,y+\overline{x}\,y+\overline{x}\,\overline{y}$	$\overline{x}+y$	$\overline{x}\,y$
12	1 1 0 0	$x\,y+x\,\overline{y}$	x	x
13	1 1 0 1	$x\,y+x\,\overline{y}+\overline{x}\,\overline{y}$	$x+\overline{y}$	$x\,\overline{y}$
14	1 1 1 0	$x\,y+x\,\overline{y}+\overline{x}\,y$	$x+y$	$x\,y$
15	1 1 1 1	$xy+x\overline{y}+\overline{x}\,y+\overline{x}\,\overline{y}$	1	0

Table 1. FND et duales des fonctions booléennes de 2 variables.

pour la fonction de la ligne 11, $f^*(x,y)=\overline{x}\,y$ à partir de la forme simplifiée de la fonction $f(x,y):=x\,y+\overline{x}\,y+\overline{x}\,\overline{y}$:

$$\begin{aligned} f(x,y) &= x\,y+\overline{x}\,y+\overline{x}\,\overline{y} \\ &= x\,y+\overline{x}\,(y+\overline{y}) \quad \text{(factorisation à gauche par } \overline{x}) \\ &= x\,y+\overline{x}\,1 \quad (y \text{ et } \overline{y} \text{ opposés}) \\ &= x\,y+\overline{x} \quad (1 \text{ est l'élément neutre pour } \times) \\ &= \overline{x}+y \quad \text{(propriété de simplification)} \end{aligned}$$

Et $\overline{x}+y$ est une FND différente de la FND f.

En revanche pour la fonction $f(x,y):=\overline{x}\,\overline{y}$ en ligne 1, l'expression de sa duale n'étant pas sous FND, on ne peut pas conclure immédiatement. C'est le cas des 4 mintermes aux lignes 1, 2, 4 et 8, mais il suffit de choisir les valeurs x et y de manière à ce que les deux littéraux soient opposés pour assurer que $f(x,y)=0$ et $f^*(x,y)=1$. On pouvait également remarquer que ces formes duales apparaissaient dans les formes simplifiées des fonctions aux lignes 7, 11, 13 et 14 respectivement.

En conclusion, les 4 fonctions autoduales sont les deux projections et leurs négations.

(5) Pour simplifier les écritures dans la suite, on note $x:=(x_1,\dots,x_n)$ et $\overline{x}=(\overline{x_1},\dots,\overline{x_n})$. La propriété (1) montre que dans la table de vérité d'une fonction autoduale f, chacune des 2^n valeurs $f(x)$ est égale à $\overline{f(\overline{x})}$. L'algorithme en découle immédiatement : on parcourt les 2^n valeurs $f(x)$ dans la table de vérité T tant que $f(x)=\overline{f(\overline{x})}$. Si toutes les égalités ont pu être vérifiées, la fonction est autoduale, sinon elle ne l'est pas.

Le pseudo-code de l'algorithme ci-dessous renvoie *Vrai* si la fonction est autoduale, *Faux* sinon. Par définition, la table de vérité d'une fonction booléenne est indexée par les entiers $i\in[\![0,2^n-1]\!]$ dont l'écriture binaire sur n bits fixe les composantes de x. On note $\neg i$ l'entier dont l'écriture binaire est la négation de celle de i :

```
ENTREE : T table de vérité de f
SORTIE : Vrai si f = f* / Faux sinon
DEBUT
··· i ← 0
··· moitie ← 2^(n-1)
··· TANT QUE ((i < moitie) ET (T[i] = ¬T[¬i])):
····· i ← i + 1
··· RENVOYER (i >= moitie)
FIN
```

NB. L'écriture binaire sur n chiffres de l'entier 2^n-1 est une succession de n chiffres 1 et comme chaque chiffre de $\neg i$ est la négation du chiffre correspondant de i, on a donc

$$\forall i\in[\![0,2^n-1]\!] \quad i+\neg i=2^n-1.$$

Par exemple pour $n=8$ bits, si $i=37=00100101_2$ alors $\neg i=218=11011010_2$ et $i+\neg i=255=11111111_2$.

Par conséquent $\neg i=(2^n-1)-i$. On en déduit que si $T[i]=f(x)$ alors $T[\neg i]=T[(2^n-1)-i]=f(\overline{x})$. Ainsi, quand on balaie la table de vérité de gauche à droite pour décrire les n-uplets booléens x, leurs négations $\overline{x}$ sont parcourues en sens inverse. On peut donc s'arrêter à la moitié de la table en fixant `fin ← 2^(n-1)`. De cette manière on évite de tester inutilement

chaque égalité $f(x) = f(\overline{x})$ deux fois. La table 2 illustre le processus de vérification (et un procédé de construction d'une fonction autoduale à la question suivante).

(6) Pour construire une fonction autoduale, il suffit de choisir les 2^{n-1} premières valeurs de la table de vérité arbitrairement, les 2^{n-1} valeurs suivantes sont fixées par leurs négations. Il y a donc $2^{2^{n-1}}$ façons possibles de fixer ces 2^{n-1} premières valeurs, ce qui nous donne $2^{2^1} = 4$ fonctions autoduales de 2 variables.

Exemple : la table de vérité d'une fonction de deux variables contient 4 valeurs, on peut donc choisir les deux premières pour $i = 0$ puis $i = 1$, par exemple 0 et 1, ce qui fixe les valeurs des deux dernières $i = (2^2-1)-0 = 3$ puis $i = (2^2-1)-1 = 2$ respectivement à 1 et 0 :

	$f(x) \rightarrow$		$\leftarrow f(\overline{x})$		
$i : \neg i$	0 : 3	1 : 2	2 : 1	3 : 0	
$x = (x_1, x_2)$	(0,0)	**(0,1)**	(1,0)	**(1,1)**	$\overline{x} = (\overline{x_1}, \overline{x_2})$
$T[i] = f(x)$	0	**1**	0	**1**	$T[\neg i] = f(\overline{x})$

TABLE 2. Vérification ou construction de la table de vérité de la fonction autoduale $f(x_1, x_2) := x_2$.

Les deux couples $(\boldsymbol{x}, \boldsymbol{y})$ tels que $f(x, y) = 1$ dans la table déterminent les mintermes de la FND de f :

$$\begin{aligned} f(x_1, x_2) &= \overset{\mathbf{0}}{\overline{x_1}}\, \overset{\mathbf{1}}{x_2} + \overset{\mathbf{1}}{x_1}\, \overset{\mathbf{1}}{x_2} \\ &= (\overline{x_1} + x_1)\, x_2 \\ &= x_2 \end{aligned}$$

C'est la fonction à la ligne 10 dans la table 1.

EXERCICE 2. Soit $\mathscr{A} := \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$ l'ensemble des lettres de l'alphabet latin et $\mathscr{B} := \{0, 1\}$ l'ensemble des booléens. On considère $V : \mathscr{A} \to \mathscr{B}$ l'application définie par

$$V(\ell) := \begin{cases} 1 & \text{si } \ell \text{ est une voyelle} \\ 0 & \text{si } \ell \text{ est une consonne.} \end{cases}$$

(1) (0.5) L'application V est-elle injective, surjective, bijective ?

(2) (0.5) Calculez les ensembles $\mathscr{C} := V^{-1}(\{0\})$ et $\mathscr{V} := V^{-1}(\{1\})$. Montrez qu'ils forment une partition de $\mathscr{A}$.

On note $\preceq$ l'ordre alphabétique défini sur $\mathscr{A}$ et $\theta : \mathscr{A} \to [\![0, 25]\!]$ l'application qui associe à une lettre x sa position $\theta(x)$ dans l'ordre alphabétique, i.e. $\theta(\mathtt{a}) = 0$, $\theta(\mathtt{b}) = 1$, …, $\theta(\mathtt{z}) = 25$.

Soit $H : \mathscr{A}^2 \to \mathscr{A}$ l'application définie par

$$H(k, \ell) = \theta^{-1}\big((\theta(k) + \theta(\ell)) \bmod 26\big)$$

où $a \bmod b$ désigne le reste de la division euclidienne de a par b.

(3) (0.5) Calculez $H(\mathtt{a}, \mathtt{b})$ et $H(\mathtt{b}, \mathtt{e})$.

(4) (0.5) La fonction H est-elle injective, surjective, bijective ?

(5) (1.0) Écrivez la fonction Python `Hacher` à deux paramètres de type `char` qui calcule la fonction H. Indication : les fonctions `ord()` et `chr()` du Python renvoient respectivement l'unicode du caractère passé en paramètre et le caractère dont l'unicode est passé en paramètre. Les unicodes de deux lettres consécutives sont des entiers consécutifs.

(6) (1.5) On considère la relation binaire $\mathscr{R}$ définie sur $\mathscr{A}^2$ par

$$(k, \ell)\, \mathscr{R}\, (k', \ell') \Leftrightarrow (k \preceq k') \wedge (\ell \preceq \ell').$$

Montrez que $\mathscr{R}$ est une relation d'ordre.

(7) (1.0) La relation d'ordre $\mathscr{R}$ est-elle totale ou partielle ?

(8) (1.5) On considère la relation binaire $\mathscr{S}$ définie sur $\mathscr{A}^2$ par

$$(k, \ell)\, \mathscr{S}\, (k', \ell') \Leftrightarrow (V(k) = V(k')) \wedge (V(\ell) = V(\ell'))$$

Montrez que $\mathscr{S}$ est une relation d'équivalence.

(9) (1.0) Quelles sont les différentes classes d'équivalence de $\mathscr{S}$?

Solution. **(1)** L'application V n'est pas injective, toutes les lettres du même type (consonne/voyelle) ayant la même image (0/1). Un seul contre exemple suffisait pour le prouver, par exemple $V(\mathtt{a}) = V(\mathtt{e})$, par conséquent elle ne peut pas être bijective. En revanche elle est surjective, puisque tous les éléments de $\mathscr{B} = \{0, 1\}$ admettent au moins un antécédent, par exemple `a` pour 1 puisque $V(\mathtt{a}) = 1$ et `b` pour 0 puisque $V(\mathtt{b}) = 0$.

(2) Pas de grand mystère dans cette question, les images réciproques des singletons $\{0\}$ et $\{1\}$ sont respectivement le sous-ensemble des consonnes et le sous-ensemble des voyelles de l'alphabet $\mathscr{A}$:

$$\mathscr{C} = V^{-1}(\{0\}) = \{\mathtt{b}, \mathtt{c}, \ldots, \mathtt{x}, \mathtt{z}\}$$
$$\mathscr{V} = V^{-1}(\{1\}) = \{\mathtt{a}, \mathtt{e}, \mathtt{i}, \mathtt{o}, \mathtt{u}, \mathtt{y}\}$$

Aucun de ces deux ensembles n'est vide, leur intersection est vide et leur réunion est l'alphabet $\mathscr{A}$ tout entier, la famille $\{\mathscr{C}, \mathscr{V}\}$ forme donc une partition de $\mathscr{A}$.

(3) On a $\theta(\mathtt{a}) = 0$, $\theta(\mathtt{b}) = 1$ et $\theta(\mathtt{e}) = 4$, par conséquent

$$\begin{aligned} H(\mathtt{a},\mathtt{b}) &= \theta^{-1}((0+1) \bmod 26) \\ &= \theta^{-1}(1) \\ &= \mathtt{b} \end{aligned} \qquad \begin{aligned} H(\mathtt{b},\mathtt{e}) &= \theta^{-1}((1+4) \bmod 26) \\ &= \theta^{-1}(5) \\ &= \mathtt{f} \end{aligned}$$

(4) L'addition dans $\mathbb{N}$ étant commutative, pour tout couple de lettres (k, ℓ), on a l'égalité $H(k, \ell) = H(\ell, k)$. Il suffit donc de prendre deux lettres k et ℓ distinctes pour montrer que H n'est pas injective, par exemple $H(\mathtt{a},\mathtt{b}) = 1 = H(\mathtt{b},\mathtt{a})$. Par conséquent, H ne peut pas être bijective. En revanche H est surjective, en effet toute lettre $\ell \in \mathscr{A}$ admet au moins l'antécédent $(\mathtt{a}, \ell) \in \mathscr{A}^2$ puisque $\theta(\mathtt{a}) = 0$:

$$\forall \ell \in \mathscr{A} \;\; H(\mathtt{a}, \ell) = \ell.$$

(5) Il fallait penser à translater les valeurs de manière à ce que le code de `a` soit aligné avec 0, ce qui n'est pas le cas avec la fonction `ord` du Python. La solution ci-dessous est volontairement décomposée :

```
def Hacher(k,l):
    decalage = ord("a")
    ok = ord(k) - decalage
    ol = ord(l) - decalage
    return chr((ok + ol) % 26 + decalage)
```

(6) Vérifions que la relation $\mathscr{R}$ est réflexive, antisymétrique et transitive. Dans les trois cas, le résultat découle directement de ces mêmes propriétés satisfaites par la relation d'ordre alphabétique, ce que nous allons détailler.

Réflexivité : il faut démontrer l'assertion

$$\forall (k, \ell) \in \mathscr{A}^2 \;\; (k, \ell) \mathrel{\mathscr{R}} (k, \ell).$$

qui est satisfaite car $k \preceq k$ et $\ell \preceq \ell$.

Antisymétrie : il faut démontrer l'assertion

$$\forall (k, \ell, k', \ell') \in \mathscr{A}^4 \;\; \big((k, \ell) \mathrel{\mathscr{R}} (k', \ell')\big) \wedge \big((k', \ell') \mathrel{\mathscr{R}} (k, \ell)\big) \Rightarrow \big((k = k') \wedge (\ell = \ell')\big).$$

D'après l'hypothèse, on a

$$\big((k \preceq k') \wedge (\ell \preceq \ell')\big) \wedge \big((k' \preceq k) \wedge (\ell' \preceq \ell)\big)$$

et par associativité et commutativité du connecteur de disjonction $\wedge$, on obtient

$$\underbrace{\big((k \preceq k') \wedge (k' \preceq k)\big)}_{\Rightarrow k = k'} \wedge \underbrace{\big((\ell \preceq \ell') \wedge (\ell' \preceq \ell)\big)}_{\Rightarrow \ell = \ell'}$$

et la relation d'ordre alphabétique étant antisymétrique, on en déduit $k = k'$ et $\ell = \ell'$.

Transitivité : il faut démontrer l'assertion

$$\forall (k, k', k'', \ell, \ell', \ell'') \in \mathscr{A}^6 \;\; \big((k, \ell) \mathrel{\mathscr{R}} (k', \ell')\big) \wedge \big((k', \ell') \mathrel{\mathscr{R}} (k'', \ell'')\big) \Rightarrow \big((k, \ell) \mathrel{\mathscr{R}} (k'', \ell'')\big).$$

D'après l'hypothèse, on tire

$$\big((k \preceq k') \wedge (\ell \preceq \ell')\big) \wedge \big((k' \preceq k'') \wedge (\ell' \preceq \ell'')\big)$$

et par associativité et commutativité du connecteur de disjonction $\wedge$, on obtient

$$\underbrace{\big((k \preceq k') \wedge (k' \preceq k'')\big)}_{\Rightarrow k \preceq k''} \wedge \underbrace{\big((\ell \preceq \ell') \wedge (\ell' \preceq \ell'')\big)}_{\Rightarrow \ell \preceq \ell''}$$

et par transitivité de l'ordre alphabétique, on obtient $k \preceq k''$ et $\ell \preceq \ell''$ ce qui nous permet de conclure que $((k, \ell) \mathrel{\mathscr{R}} (k'', \ell''))$.

C'est une relation d'ordre partielle, en effet on ne peut pas comparer tous les couples de lettres de $\mathscr{A}$, entre autres les couples $(\mathtt{a},\mathtt{b})$ et $(\mathtt{b},\mathtt{a})$.

(7) Vérifions que la relation $\mathscr{S}$ est réflexive, symétrique et transitive. Dans les trois cas, le résultat découle directement de ces mêmes propriétés satisfaites par la relation binaire d'égalité, ce que nous allons détailler.

Réflexivité : il faut démontrer l'assertion

$$\forall (k,\ell) \in \mathscr{A}^2 \quad (k,\ell)\,\mathscr{S}\,(k,\ell).$$

qui est satisfaite car $V(k) = V(k)$ et $V(\ell) = V(\ell)$.

Symétrie : il faut démontrer l'assertion

$$\forall (k,k',\ell,\ell') \in \mathscr{A}^4 \quad \big((k,\ell)\,\mathscr{S}\,(k',\ell')\big) \Rightarrow \big((k',\ell')\,\mathscr{S}\,(k,\ell)\big)\,.$$

D'après l'hypothèse, on tire $V(k) = V(k')$ et $V(\ell) = V(\ell')$ ce qui prouve par symétrie de l'égalité que $(k',\ell')\,\mathscr{S}\,(k,\ell)$.

Transitivité : il faut démontrer l'assertion

$$\forall (k,k',k'',\ell,\ell',\ell'') \in \mathscr{A}^6 \ \big((k,\ell)\,\mathscr{S}\,(k',\ell')\big) \wedge \big((k',\ell')\,\mathscr{S}\,(k'',\ell'')\big) \Rightarrow \big((k,\ell)\,\mathscr{S}\,(k'',\ell'')\big)\,.$$

D'après l'hypothèse, on tire

$$\big(\big(V(k) = V(k')\big) \wedge \big(V(\ell) = V(\ell')\big)\big) \wedge \big(\big(V(k') = V(k'')\big) \wedge \big(V(\ell') = V(\ell'')\big)\big)$$

Par associativité et commutativité du connecteur de disjonction $\wedge$ puis par transitivité de l'égalité on obtient $V(k) = V(k'')$ et $V(\ell) = V(\ell'')$ puis finalement $(k,\ell)\,\mathscr{S}\,(k'',\ell'')$.

(8) Deux couples de lettres sont donc en relation si leurs projections respectives sont de même nature (consonne ou voyelle). Il y a donc 4 classes possibles, les couples de consonnes, les couples de voyelles, les couples dont la première projection est une consonne et la seconde une consonne et réciproquement. On sait que les classes d'équivalence d'une relation d'équivalence forment une partition de l'ensemble, on a donc

$$\mathscr{A} \times \mathscr{A} = (\mathscr{V} \times \mathscr{V}) \sqcup (\mathscr{C} \times \mathscr{C}) \sqcup (\mathscr{C} \times \mathscr{V}) \sqcup (\mathscr{V} \times \mathscr{C}).$$