

## Logique Booleene

### Definitions et props

Définition 1: Commutatif les variables peuvent etre inverses

Définition 2: L'arbre de Derivation C'est un format de pour represente une proposition

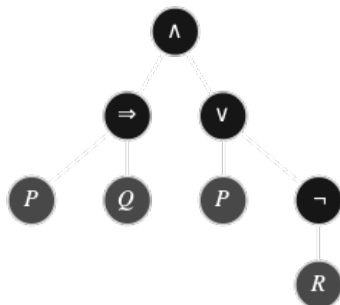


Figure 1:  $(P \Rightarrow Q) \wedge (P \vee \neg R)$

Définition 3: Loi de De Morgan Soit  $P$  et  $Q$  deux assertions, alors  
 $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$   
 $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$

### Tables de verite

il est assume qu'un connecteur est commutatif sauf mentionne autrement

table de  $\wedge$ , dit conjonction, lu "et": q binaire

$\perp$	$\perp$	$\perp$
$\perp$	$\top$	$\perp$
$\top$	$\perp$	$\perp$
$\top$	$\top$	$\top$

table de  $\vee$  dit disjonction, lu "ou": q binaire

$\perp$	$\perp$	$\perp$
$\perp$	$\top$	$\top$
$\top$	$\perp$	$\top$
$\top$	$\top$	$\top$

Définition 4: Clause on dit clause conjonctive (ou respectivement disjontive) toute formule composé de conjonction (respectivement disjonction)

table de  $\oplus$ : q binaire

$\perp$	$\perp$	$\perp$
$\perp$	$\top$	$\top$
$\top$	$\perp$	$\top$
$\top$	$\top$	$\perp$

table de  $\Rightarrow$ : q binaire dit non commutatif

$\perp$	$\perp$	$\top$
$\perp$	$\top$	$\top$
$\top$	$\perp$	$\perp$
$\top$	$\top$	$\top$

autrement dit, vrai sauf si  $p$  est vrai et  $q$  est faux

table de  $\Leftrightarrow$ : q binaire

$\perp$	$\perp$	$\top$
$\perp$	$\top$	$\perp$
$\top$	$\perp$	$\perp$
$\top$	$\top$	$\top$

vrai si les deux variables ont la meme valeur

### Proprietes

- commutativite de  $\wedge$  et  $\vee$

$$(p \wedge q) \equiv (q \wedge p)$$

$$(p \vee q) \equiv (q \vee p)$$

- associativite de  $\wedge$  et  $\vee$

$$((p \wedge q) \wedge r) \equiv ((q \wedge r) \wedge p) \quad ((p \vee q) \vee r) \equiv ((q \vee r) \vee p)$$

- idempotence de  $\wedge$  et  $\vee$

$$(p \wedge p) \equiv p$$

$$(p \vee p) \equiv p$$

### Modus{ponens, tollens}

Définition 5: modus ponens Soit  $P$  et  $Q$  deux propositions, si  $P$  est vrai et  $P \Rightarrow Q$  est vrai, alors  $Q$  est vrai.

$$P, P \Rightarrow Q \vdash Q$$

car  $P \Rightarrow q \equiv \neg Q \Rightarrow \neg P$

Définition 6: modus tollens Soit  $P$  et  $Q$  deux propositions, si  $\neg Q$  est vrai et  $P \Rightarrow Q$  est vrai, alors la proposition  $\neg P$  est vrai.

$$\neg Q, P \Rightarrow Q \vdash \neg P$$

### TPs

Question 1: Ecrire une fonction

`interpretations(nbVar)` qui renvoie le tuple constitue de toutes les interpretations possible de nbvar variables propositionnelles

la technique que j'ai opte est de calculer tous les nombre possible en binaire jusqu'a  $2^{nbvar}$ , puis de les retranscrire en tuple de vrai/faux. Voici le code (on assume une fonction translate to tuple defini comme le suit)

# Q1: ecrire une fonction Inter(nbvar) qui renvoie le

tuple constitue de toutes les interpretations possible de nbvar variables propositionnelles

```
def translateTuple(binary: str):
    result = []
    for i in binary:
        if i == '1':
            result.append(True)
        else:
            result.append(False)

    return tuple(result)

def inter(nbvar):
    finalresult = ()
    for i in range(nbvar**2):
        result = bin(i)
        result = result[2:]
        while len(result) < nbvar:
            result = '0' + result
        result = translateTuple(result)
        finalresult += result,
    return finalresult
```

## Axiomes et predicats

### Ensembles

**Définition 7: ensemble** Un ensemble est une collection  $X$  d'objets definis et unique. un objet appartenant a l'ensemble est dit membre de  $X$  et on dit que l'objet est membre. un membre est unique dans un ensemble, il ne peut pas y avoir deux fois le meme element

exemple:

$$\{a, b, c, a\} = \{a, b, c\}$$

sur python, un type ensemble existe qui est appele `set`

**Définition 8: Difference** Soit  $X$  et  $Y$  deux ensembles. la difference entre les ensembles  $X$  et  $Y$  est l'ensemble  $\{x \in X \mid x \notin Y\}$ , qui est l'ensemble qui contient les elements de  $X$  mais pas les elements de  $Y$ . on note aussi  $X \setminus Y$  l'ensemble qui contient seulement les differences d'un ensemble  $X \cap Y$  est  $X \Delta Y$

**Définition 9: Cardinal** On appelle le cardinal d'un ensemble sa taille. Lorsqu'un ensemble est fini, le cardinal est la longueur de cette ensemble

### Predicats

**Définition 10: Predicat** enonce contenant des variables tel qu'en substituant chaque variables par une valeur choisi, on obtient une proposition

exemple:  $x|P(x)$  (se lit  $x$  tel que  $P(x)$ ) est un predicat dans lesquelles la proposition  $P(x)$  est vraie pour  $x$  la theorie de ZF distingue deux types de predicats:

1. **predicat collectivisant**: un predicat  $P(X)$  tel que les valeurs de  $x$  pour lesquelles la proposition  $P(x)$  est vrai constituent un ensemble note  $\{x|P(x)\}$
2. **predicat non collectivisant**: un predicat  $P(x)$  tel que les valeurs  $x$  pour lesquelles la prop  $P(x)$  est vraie ne constituent pas un ensemble

considerant le predicat  $P(x, y)$  defini sur deux variables reelles  $x$  et  $y$  suivant:

$$x^2 - y = 1$$

on peut definir le predicat  $Q(x)$  de la variable suivante:

$$\exists y \in \mathbb{R} x^2 - y = 1$$

### Quantificateurs

**Définition 11: quantificateur** Il existe 3 quantificateurs:

- $\forall$  qui se lit "pour tout" (appele forall en latex et typst)
- $\exists$  qui se lit "il existe"
- $\exists!$  qui est un "il existe" unique

le quantificateur  $\exists!$  est lui meme une proposition qui est:  $(\exists x \in X P(x)) \wedge (\forall x \in X \forall y \in X P(x) \wedge P(y) \Rightarrow x = y)$  le terme de gauche code l'existence et le terme droit l'unicite en exprimant sous forme contraposee que deux elements distincts  $x$  et  $y$  de l'ensemble  $X$  ne peuvent simultanement satisfaire le predicat  $P(x) : x \neq y \Rightarrow \neg(P(x) \wedge P(y))$ .

### Axiomes

**Définition 12: axiome de l'inclusion** Soit  $X$  et  $Y$  deux ensembles. on dit que  $X$  est inclus dans  $Y$  ou que  $X$  est une partie de  $Y$  ou encore que  $X$  est un sousensemble de  $Y$ , ce que l'on note  $X \subseteq Y$  ou  $Y \supseteq X$  seulement si  $\forall x x \in X \Rightarrow x \in Y$

**Définition 13: axiome d'extension**

Soit  $X$  et  $Y$  deux ensembles, alors  $X = Y$  si et seulement si

$$(X \subseteq Y) \wedge (Y \subseteq X)$$

**Définition 14: axiome de la paire** soit  $a, b$  deux objet. le predicat  $(x = a) \vee (x = b)$  est collectivisant en  $x$ . l'ensemble definit est  $\{a, b\}$

$$\{x \mid (x = a) \vee (x = b)\}$$

### Caclul Booleen

#### Algre de boole

soit  $\mathbb{B}$  un ensemble munit d'une structure algebrique, on l'appelle algre de boole.

**Définition 15:** on appelle booleen toute variable defini sur un ensemble a deux elements

Pour simplifier l'écriture des expressions logique, l'operande  $\neg$  peut etre ecrit de cette facon:  $\bar{x}$ . et on a

$x$	0	1
$\bar{x}$	1	0

dans le cadre de l'algebre de Boole, un litteral designe la aussi une variable  $x$  (litteral positif) ou sa negation  $\bar{x}$  (litteral negatif)

Proprietes de calcul

on dispose des nombreuses proprietess suivantes heritees du calcul propositionnel:

1. associativite:  $(a + b) + c = a + (b + c) = a + b + c$
2. commutativite  $a + b = b + a$
3. distributivite  $a(b + c) = ab + (ac)$
4. idempotence:  $a + a + a + a \dots = a$  et  $aaa \dots = a$
5. element neutre:  $a + 0 = 0 + a = a$  et  $a1 = 1a = a$
6. absorption  $0a = a$  et  $1 + a = 1$
7. simplification:  $a + \bar{a}b = a + b$  et  $a(\bar{a} + b) = ab$
8. redondance:  $ab + \bar{a}c = ab + \bar{a}c + bc$  et  $(a + b)(\bar{a} + c) = (a + b)(\bar{a} + c)(b + c)$
9. DeMorgan:  $\overline{ab} = \bar{a} + \bar{b}$
10. Involution:  $\bar{\bar{a}} = a$
11. tiers exclu:  $\bar{a} + a = 1$
12. non contradiction:  $a\bar{a} = 0$

on retrouve les cinq autres operateur binaire, implication, equifvvalence, disjonction exclusive, non conjonction et non disjonction:

$$\begin{aligned} a \Rightarrow b &= \bar{a} + b, \\ a \Leftrightarrow b &= (\bar{a} + b)(a + \bar{b}) \\ a \oplus b &= (a + b)(\bar{a} + \bar{b}) \\ a \uparrow b &= \bar{a}\bar{b} \\ a \downarrow b &= \widetilde{a + b} \end{aligned}$$

qui ont les tables de verite:

$\Rightarrow$	0	1
0	1	1
1	0	1

$\Leftrightarrow$	0	1
0	1	0
1	0	1

$\oplus$	0	1
0	0	1
1	1	0

$\uparrow$	0	1
0	1	1
1	1	0

$\downarrow$	0	1
0	1	0
1	0	0

**Definitions:**

**Définition 16: antilogie** L'antilogie est le cas ou une formule repond toujours faux, a l'inverse de la tautologie qui repond toujours vrai

**Code Gray**

**Définition 17: Code Gray**

**Minterme, maxterme**

**Définition 18: Minterme, Maxterme** on appelle Minterme toute fonction d'ordre  $n$ , prenant une seule fois la valeur 1

**Relations et applications**  
analyse combinatoire

**Ensembles naturel**

**Définition 19: Ensemble Naturel** On appelle ensemble naturel  $(N, \preceq)$  tout ensemble ordonne qui satisfait les trois proprietes suivantes:

- Toute partie non vide admet un plus petit element
- Toute partie non vide et majoree admet un plus grand element
- L'ensemble n'admet pas de plus grand element

l'existence d'un ensemble naturel est acquise grace a l'axiome de l'infini (consulter wikipedia) Pour demontrer qu'un ensemble naturel est ordonne, on peut emettre la proposition suivante:

$$\exists m \in \{a, b\} (m \preceq a) \wedge (m \preceq b)$$

deux element  $a, b$  dans l'ensemble  $N$ . D'apres l'axiome de la paire, l'ensemble  $\{a, b\}$  existe, n'est pas vide et admet donc un plus petit element (un ensemble naturel est toujours minore mais jamais majore) Comme  $m \in N$  on a  $(m = a) \wedge (m = b)$  et on deduit que  $(a \preceq b) \vee (b \preceq a)$

Soit  $n \in \mathbb{N}$  la demi droite  $]n, \rightarrow[$  n'est pas vide, sinon  $n$  serait le plus grand element, ce qui va a l'encontre de la 3eme propriete.  $]n, \rightarrow[$  admet un plus petit element appele  $\text{succ}(n)$ , le successeur de  $n$

## recurrence

**Définition 20:** Theoreme principe de recurrence  
Toute partie de  $\mathbb{N}$  qui contient 0 et stable pour l'application successeur est egale a  $\mathbb{N}$

**Définition 21:** theoreme recurrence simple  
Soit  $P(n)$  un predicat sur  $\mathbb{N}$  et  $a \in \mathbb{N}$ . Si les deux propositions sont satisfaites:  
•  $P(a)$  init  
•  $\forall n \in \mathbb{N} P(n) \Rightarrow P(n+1)$  heredite  
alors  $\forall n \in \llbracket a, \rightarrow \rrbracket P(n)$

**Définition 22:** theoreme recurrence forte  
Soit  $P(n)$  un predicat sur  $\mathbb{N}$  et  $a \in \mathbb{N}$ . Si les deux propositions sont satisfaites:  
•  $P(a)$  init  
•  $\forall n \in \mathbb{N} (\forall k \in \llbracket a, n \rrbracket P(k)) \Rightarrow P(n+1)$  heredite forte  
alors  $\forall n \in \llbracket a, \rightarrow \rrbracket P(n)$

**Définition 23:** theoreme recurrence multiple  
Soit  $P(n)$  un predicat sur  $\mathbb{N}$  et  $a \in \mathbb{N}$ . Si les deux propositions sont satisfaites:  
•  $\forall i \in \llbracket 0, k-1 \rrbracket P(a+i)$  init  
•  $\forall n \in \mathbb{N} P(n) \Rightarrow P(n+k)$   
alors  $\forall n \in \llbracket a, \rightarrow \rrbracket P(n)$

**Définition 24:** theoreme recurrence finie  
Soit  $P(n)$  un predicat sur  $\mathbb{N}$  et  $a \in \mathbb{N}$ . Si les deux propositions sont satisfaites:  
•  $P(a)$  init  
•  $\forall n \in \llbracket a, m-1 \rrbracket P(n) \Rightarrow P(n+1)$   
alors  $\forall n \in \llbracket a, m \rrbracket P(n)$

## Arithmetique

### Les groupes quotient de $\mathbb{Z}$

continuation du chapitre des groupes

### Divisibilite, nombres premiers

**Définition 25:** divisibilite soit  $a, b$  deux nombres, on dit que  $a$  divise  $b$  ou que  $a$  est diviseur de  $b$  ou encore que  $b$  est multiple de  $a$  si et seulement si

$$\exists c \in \mathbb{Z} ac = b$$

$a$  etant le diviseur,  $c$  etant le quotient,  
 $b$  etant le nombre

### Les anneaux $\mathbb{Z}/\mathbb{Z}_n$

#### Structure d'anneaux

L'ensemble  $\mathbb{Z}$  est muni de deux lois de composition interne, l'addition et la multiplication dont les proprietes lui confere une structure d'anneaux une structure d'anneaux

**Définition 26:** anneaux on appelle anneau tout triplet  $(A, +, *)$  ou  $+$  et  $*$  sont des lois de composition interne dite d'addition et de multiplication qui satisfont les proprietes suivantes:  
•  $(A, +)$  est un groupe commutatif dont l'element neutre est 0  
•  $(A, *)$  est un groupe associatif dont l'element neutre est 1

### Les theoremes de Gauss, Euler et Fernet

## bases et codage

## rappels

decimale	binaire	octal	hexa
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

## rappels logique

$x$	$y$	$x+y$	$xy$	$\tilde{x}$
0	0	0	0	1
0	1	1	0	1
1	0	1	0	0
1	1	1	1	0

Arithmetique tronque a gauche:

## logique combinatoire

Définition 27: tableau de Karnaugh il sert a représenter l'ensemble des arguments d'une fonction booléenne, a la meme facon qu'un tableau de valeur. cette forme est efficace pour trouver:

- la FND d'une fonction
- trouver l fonction booléenne ayant le moins de variable et d'opérateurs possible: simplification des fonctions booléennes

un tableau de karnaugh a pour argument  $n$ , qui signifie  $n$  nombre d'arguments d'une fonction booléenne

exemple pour un tableau  $n = 3$ :

xy	00	01	11	10
z				
0	$f(0,0,0)$	$f(0,1,0)$	$f(1,1,0)$	$f(1,0,0)$
1	$f(0,0,1)$	$f(0,1,1)$	$f(1,1,1)$	$f(1,0,1)$

Définition 28: forme nominal disjonctive (FND) let  $n$  variables,  $x_1...x_n$ , on appelle monome d'ordre  $n$  le produit  $y_1, y_2...y_n$  avec  $y_i = x_i$  ou  $y_i = \tilde{x}_i$  pour chaque  $i \in \{1, ..., n\}$ . une fonction est dite sous forme nominal disjonctive si la fonction est une somme de monomes d'ordre  $n$ . toute fonctions non nulle de  $n$  variables peut s'ecrire de facon unique sous forme nominale disjonctive.

exemple: soit une fonction  $f$  booléenne de 2 arguments dont son tableau de karnaugh est

xy	00	01	11	10
z				
0	1	0	1	1
1	0	1	0	0

La FND de  $f$  est  $f(x,y,z) = \tilde{x}\tilde{y}\tilde{z} + xy\tilde{z} + x\tilde{y}\tilde{z} + \tilde{x} + yz$   
on peut donc simplifier la fonction a

$$xy + x\tilde{y} = x$$

en effet  $xy + x\tilde{y} = x(y + \tilde{y}) = x1 = x$

## logique sequentielle

## Conception d'algorithme

**Définition 29:** Invalider un algo Pour montrer qu'un algo n'est pas valide, il suffit de montrer un contre exemple, soit un cas ou l'algorithme ne marcherait pas

## Analyse asymptotique

**Définition 30:** Analyser un algorithme c'est analyser les couts par rapport au temps d'execution, l'espace memoire, et la consommation electrique

**Définition 31:** le modele random access machine machine hypothetique ou:

- les operands consomment une unite de temps
- les boucles depend du nombre d'iterations et des operation inside
- un read consomme une unite de temps
- la memoire est illimite

l'efficacite d'un algo est defini par une fonction notee  $C(n)$  ou  $T(n)$ , meme si dans un cas reel ca serait plutot note  $O(n)$

exemple:

- recherche d'un element:
  - n cases a tester
  - 5 cases: > 5 tests
  - 10 cases: > 10 tests
- ramassage de plots:
  - n! chemins a tester
  - 5 plots: 120 chemins possible

la notation est qui suit:

- $\Omega(n)$ : meilleur cas
- $O(n)$ : pire cas
- $\Theta(n)$ : cas moyen

**Définition 32:**  $f(n) = O(g(n))$  il existe une constance  $c$  et un entier  $n_0$  tels que  $\forall n \geq n_0, f(n) \leq cg(n)$

exemples:

- $3n^2 - n + 6 = O(n^2)$  en prenant  $c = 3$  et  $n_0 = 6$
- $3n^2 - n + 6 = O(n^3)$  en prenant  $c = 1$  et  $n_0 = 4$
- $3n^2 - n + 6 \neq O(n)$  car  $\forall c, cn < 3n^2 - n + 6$  quand  $n > c + 1$

**Définition 33:**  $f(n) = \Omega(g(n))$  il existe une constance  $c$  et un entier  $n_0$  tels que  $\forall n \geq n_0, f(n) \geq cg(n)$

exemples:

- $3n^2 - n + 6 = \Omega(n^2)$  en prenant  $c = 2$  et  $n_0 = 2$
- $3n^2 - n + 6 \neq \Omega(n^3) \forall c, 3n^2 - n + 6 < cn^3$  quand  $cn > 3$  et  $n > 6$
- $3n^2 - n + 6 = \Omega(n)$  en prenant  $c = 1$  et  $n_0 = 1$

**Définition 34:**  $f(n) = \Theta(g(n))$  il existe une constance  $c$  et un entier  $n_0$  tels que  $\forall n \geq n_0, c_1g(n) \leq f(n) \leq c_2g(n)$   
( $f(n) = O(g(n))$  et  $f(n) = \Omega(g(n))$ )

- $3n^2 - n + 6 = \Theta(n^2)$
- $3n^2 - n + 6 = \Theta(n^3)$
- $3n^2 - n + 6 = \Theta(n)$

## Bases d'algo

### parcours de tableau

### Algos de recherche

#### Algos de tri

3 types d'algos de tri:

- tri par selection
- tri par propagation
- tri par insertion

tout les algorithmes reposent sur une méthode d'echange d'items basé sur leurs indice:

```
def swap(T, i, j):
    a = T[i]
    T[i] = T[j]
    T[j] = a
```

#### tri par selection

bien que simple, l'algorithme est considere comme inefficace a cause de son temps d'execution quadratique

$$\Omega(n) = n^2$$

$$O(n) = n^2$$

$$\Theta(n) = n^2$$

il consiste a parcourir une liste, et echanger le plus petit element par le premier, puis d'avancer l'indice du premier jusqu'a finir de parcourir la liste

```
i ← 1
while i < length(A)
    j ← i
    while j > 0 and A[j-1] > A[j]
        swap A[j], A[j-1]
        j ← j - 1
    end while
    i ← i + 1
end while
```

#### tri par propagation (Bubble sort)

il a une complexite de  $n^2$ , sauf pour le meilleur cas, ou  $\Omega(n) = n$  il consiste a echanger les elements qui sont dans le désordre ( $n+1 < n$ ), a la fin de chaque iteration, le dernier element est le plus grand, donc l'indice est soustre a chaque fois

```
tri_à_bulles(Tableau T)
    pour i allant de (taille de T)-1 à 1
        pour j allant de 0 à i-1
            si T[j+1] < T[j]
                (T[j+1], T[j]) ← (T[j], T[j+1])
```

#### tri par insertion

il parcourt la liste, et a chaque fois que l'algo trouve un nombre inferieur au precedent, il revient en arriere pour le placer correctement

```
procédure tri_insertion(tableau T)

    pour i de 1 à taille(T) - 1

        # mémoriser T[i] dans x
        x ← T[i]

        # décaler les éléments T[0]..T[i-1] qui sont
plus grands que x, en partant de T[i-1]
        j ← i
        tant que j > 0 et T[j - 1] > x
            T[j] ← T[j - 1]
            j ← j - 1

        # placer x dans le "trou" laissé par le
décalage
        T[j] ← x
```

## Algos de recherche

### Recherche Dichotomique

Soit une liste triée, l'algorithme consiste à chercher  $x$  dans la moitié de la liste:

- si  $T[i] < x$ , on prend la partie gauche
- si  $T[i] > x$ , on prend la partie droite

En code, cela consiste à prendre l'indice du milieu,  $n/2$  pour un tableau de longueur  $n$ , puis de comparer la taille. Enfin d'avancer l'indice de 1 si le nombre trouvé est petit, et l'inverse si il est trop grand.

La dichotomie est aussi utilisée pour la recherche de pic:

```
que (b - a) > ε
    m ← (a + b) / 2
    Si (f(a)*f(m) ≤ 0) alors
        b ← m
    sinon
        a ← m
    Fin Si
Fin Tant que
```

## Piles et Files

### Piles

Les piles (ou stack) sont des structures de données