Lògica en la Informàtica
Definició de Lògica Proposicional (LProp)
Deducció en Lògica Proposicional

José Miguel Rivero

Dept. Ciències de la Computació
Universitat Politècnica de Catalunya (UPC)

Primavera 2025

Crèdits

El material utilitzat en aquesta presentació ha estat extret del elaborat pel professor Robert Nieuwenhuis (Dept. CS, UPC) per l'assignatura *Lògica en la Informàtica* de la FIB.

En particular, del llibre *Lógica para informáticos* - Farré, R. [et al.], Marcombo, 2011. ISBN: 9788426716941.

Temari

Exercicis del capítol 2 dels apunts: ☞ `p2.pdf`

1. Exercici 26 [cost de SAT]
2. Exercici 27 [funcions booleanes]
3. Exercici 28 [funcions booleanes]
4. Exercici 31 [funcions booleanes]
5. Exercici 32 [funcions booleanes]
6. Exercici 37 [simplificació de codi]
7. Exercici 33 [formalització i raonament]
8. Exercici 34 [altres lògiques]
9. Exercici 35 [altres lògiques]

Capítol 3 dels apunts: ☞ `p3.pdf`

Exercici 26

26. (dificultat 2) Suposem que $|\mathcal{P}| = 100$ i que ens interessa determinar si una fórmula F construïda sobre $\mathcal{P}$ és satisfactible o no. Si l'algorisme està basat en una anàlisi de la taula de veritat, i avaluar F en una interpretació I donada costa un microsegon (10^{-6} segons), quants anys trigarà?
Més endavant veurem tècniques que moltes vegades funcionen millor.

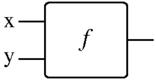
Quantes interpretacions hi ha? hi ha 2^{100}

$$\begin{aligned} 2^{10} = 1024 &\approx 10^3 \\ 2^{100} &\approx 10^{30} \end{aligned}$$

Avaluar-les totes trigarà $2^{100} \cdot 10^{-6}$ segons $\approx 10^{30} \cdot 10^{-6}$ segons $\approx 10^{24}$ segons $\approx 10^{24}/(365 \cdot 24 \cdot 3600)$ anys $\approx 4 \cdot 10^{16}$ anys aprox.

Exercici 27

27. (dificultat 2) Una funció booleana de n entrades és una funció $f : \{0,1\}^n \to \{0,1\}$, és a dir, una funció que prem com entrada una cadena de n bits i retorna un bit. Quantes funcions booleanes de n entrades hi ha?



Hi ha 2^{2^n}: 2 elevat a (2 elevat a n): hi ha tantes funcions com tires de 2^n bits

Exercici 27

27. (dificultat 2) Una funció booleana de n entrades és una funció $f : \{0,1\}^n \to \{0,1\}$, és a dir, una funció que prem com entrada una cadena de n bits i retorna un bit. Quantes funcions booleanes de n entrades hi ha?

Hi ha 2^{2^n}: 2 elevat a (2 elevat a n): hi ha tantes funcions com tires de 2^n bits

Exemple $n = 2$: tantes funcions com tires de 2^n bits = tires de 4 bits = 2^4

x	y	0	and	$\neg(x \to y)$	x	$\neg(y \to x)$	y	xor	or	$\cdots$
0	0	0	0	0	0	0	0	0	0	
0	1	0	0	0	0	1	1	1	1	
1	0	0	0	1	1	0	0	1	1	
1	1	0	1	0	1	0	1	0	1	

x	y	$\cdots$	nor	$=$	$\neg y$	$y \to x$	$\neg x$	$x \to y$	nand	1
0	0		1	1	1	1	1	1	1	1
0	1		0	0	0	0	1	1	1	1
1	0		0	0	1	1	0	0	1	1
1	1		0	1	0	1	0	1	0	1

Exercici 27

27. (dificultat 2) Una funció booleana de n entrades és una funció $f : \{0,1\}^n \to \{0,1\}$, és a dir, una funció que prem com entrada una cadena de n bits i retorna un bit. Quantes funcions booleanes de n entrades hi ha?

Hi ha 2^{2^n}: 2 elevat a (2 elevat a n): hi ha tantes funcions com tires de 2^n bits

Exemple $n = 3$: hi ha $2^8 = 256$

x	y	z
0	0	0
0	0	1
	$\dots$	
1	1	1

Si $n = 4$, hi ha $2^{64} \approx 65000$

Exercici 28

28. (dificultat 2) Cada fórmula F representa una única funció booleana: la que retorna 1 exactament per a aquellas cadenes de bits I tals que $eval_I(F) = 1$. Per això, dues fórmules són lògicament equivalents si i només si representen la mateixa funció booleana. Quantes funcions booleanes (o quantes fórmules lògicamente no-equivalents) hi ha en funció de $n = |\mathcal{P}|$?

Per exemple, la funció booleana "and" (de 2 entrades x, y) la podem representar mitjançant les fórmules

$x \wedge y$
$(x \wedge y) \wedge y$
$\neg(\neg x \vee \neg y)$
$\neg(\neg x \vee \neg y) \wedge y$
$\ldots$

Exercici 28

28. (dificultat 2) Cada fórmula F representa una única funció booleana: la que retorna 1 exactament per a aquellas cadenes de bits I tals que $eval_I(F) = 1$. Per això, dues fórmules són lògicament equivalents si i només si representen la mateixa funció booleana. Quantes funcions booleanes (o quantes fórmules lògicamente no-equivalents) hi ha en funció de $n = |\mathcal{P}|$?

Hi ha 2^{2^n} [2 elevat a (2 elevat a n)]: hi ha tantes funcions com tires de 2^n bits

Exercici 31

31. (dificultat 3) Escriu en una taula de veritat les 16 funcions booleanes de 2 entrades. Quantes de elles només depenen d'una de las dues entrades? Quantes depenen de zero entrades? Les altres, vistes com conectives lògicas, reben algun nom?

x	y	0	and	$\neg(x \to y)$	x	$\neg(y \to x)$	y	xor	or	$\cdots$
0	0	0	0	0	0	0	0	0	0	
0	1	0	0	0	0	1	1	1	1	
1	0	0	0	1	1	0	0	1	1	
1	1	0	1	0	1	0	1	0	1	

x	y	$\cdots$	nor	$=$	$\neg y$	$y \to x$	$\neg x$	$x \to y$	nand	1
0	0		1	1	1	1	1	1	1	1
0	1		0	0	0	0	1	1	1	1
1	0		0	0	1	1	0	0	1	1
1	1		0	1	0	1	0	1	0	1

Exercici 31

31. (dificultat 3) Escriu en una taula de veritat les 16 funcions booleanes de 2 entrades. Quantes de elles només depenen d'una de las dues entrades? Quantes depenen de zero entrades? Les altres, vistes com conectives lògicas, reben algun nom? Ja sabem que podem expressar qualsevol funció booleana amb el conjunt de tres connectives $\{\wedge, \vee, \neg\}$, és a dir, qualsevol funció booleana és equivalent a una fórmula construïda sobre aquestes tres connectives. És cert això també para algún conjunto de només dues de les 16 funcions? (Hi ha diverses maneres, però basta amb donar una sola.)

Sí, amb només *or* i *not*, per exemple: $x \wedge y \equiv \neg(\neg x \vee \neg y)$

Exercici 32

32. (dificultat 3) Demostra que qualsevol funció booleana de dues entrades es pot expressar amb només *nor* o bé amb només *nand*, on $nor(F,G)$ és $\neg(F \vee G)$, i $nand(F,G)$ és $\neg(F \wedge G)$.

Ho fem per *nand*:

$$\neg F \equiv F \; nand \; F$$

$$F \vee G \equiv \neg(\neg F \wedge \neg G) \equiv \neg F \; nand \; \neg G \equiv (F \; nand \; F) \; nand \; (G \; nand \; G)$$

$$F \wedge G \equiv \neg(F \; nand \; G) \equiv (F \; nand \; G) \; nand \; (F \; nand \; G)$$

Exercici 37

37. (dificultat 3) Considera el següent fragment de codi, que retorna un booleà:

```
int i;
bool a, b;
...
if (a and i>0)         return b;      // (1)
else if (a and i<=0) return false; // (2)
else if (a or b)      return a;      // (3)
else                   return (i>0); // (4)
```

Simplifíca'l sustituïnt els valors de retorn per un sol valor de retorn que sigui una expressió booleana en $i > 0$, a i b:

```
int i;
bool a, b;
return ...;
```

Exercici 37

37. (dificultat 3) Considera el següent fragment de codi, que retorna un booleà:

```
int i;
bool a, b;
...
if (a and i>0)        return b;     // (1)
else if (a and i<=0) return false; // (2)
else if (a or b)      return a;     // (3)
else                  return (i>0); // (4)
```

$$\text{if\_then\_else}(C, F, G) \equiv (C \rightarrow F) \wedge (\neg C \rightarrow G)$$

Exercici 37

37. (dificultat 3) Considera el següent fragment de codi, que retorna un booleà:

```
int i;
bool a, b;
...
if (a and i>0)         return b;      // (1)
else if (a and i<=0)   return false;  // (2)
else if (a or b)       return a;      // (3)
else                   return (i>0);  // (4)
```

Tindrem tres símbols de predicat: a, b, $i > 0$.

a	b	$i > 0$	return	
0	0	0	0	(4)
0	0	1	1	(4)
0	1	0	0	(3)
0	1	1	0	(3)
1	0	0	0	(2)
1	0	1	0	(1)
1	1	0	0	(2)
1	1	1	1	(1)

```
return ((not a) and (not b) and i>0) or
       (a and b and i>0);
```

```
return (a==b and i>0);
```

Exercici 33

33. (dificultat 3) Tres estudiants A, B i C són acusats d'introduir un virus en les sales d'ordinadors de la FIB. Durant l'interrogatori, les declaracions són les següents:

- A diu: "B ho va fer i C és innocent"
- B diu: "Si A és culpable llavors C també ho és"
- C diu: "Jo no ho vaig fer, ho va fer almenys un dels altres dos"

a) Són les tres declaracions contradictories?

b) Assumint que tots son innocents, qui o quins van mentir en la declaració?

c) Assumint que ningún va mentir, qui és innocent i qui és culpable?

Exercici 33

33. (dificultat 3) Tres estudiants A, B i C són acusats d'introduir un virus en les sales d'ordinadors de la FIB. Durant l'interrogatori, les declaracions són les següents:

- A diu: "B ho va fer i C és innocent"
- B diu: "Si A és culpable llavors C també ho és"
- C diu: "Jo no ho vaig fer, ho va fer almenys un dels altres dos"

Introduïm símbols de predicat: a,b,c que signifiquen: "A ho va fer", "B ho va fer", "C ho va fer".

Les tres declaracions són:

- A diu: $b \wedge \neg c$
- B diu: $a \rightarrow c$
- C diu: $\neg c \wedge (a \vee b)$

Exercici 33

33. (dificultat 3) Tres estudiants A, B i C són acusats d'introduir un virus en les sales d'ordinadors de la FIB. Durant l'interrogatori, les declaracions són les següents:

Les tres declaracions són:

- A diu: $b \wedge \neg c$
- B diu: $a \rightarrow c$
- C diu: $\neg c \wedge (a \vee b)$

a) Són les tres declaracions contradictories?

Per saber si poden ser veritat les tres declaracions, formalment, hem de veure si és satisfactible la conjunció (*and*) de les tres fórmules.

Només hi ha un model I: $I(a) = 0$, $I(b) = 1$, $I(c) = 0$.
Per tant, no són contradictòries.

Exercici 33

33. (dificultat 3) Tres estudiants A, B i C són acusats d'introduir un virus en les sales d'ordinadors de la FIB. Durant l'interrogatori, les declaracions són les següents:

Les tres declaracions són:

- A diu: $b \wedge \neg c$
- B diu: $a \to c$
- C diu: $\neg c \wedge (a \vee b)$

b) Assumint que tots son innocents, qui o quins van mentir en la declaració?

Si tots són innocents, A i C van mentir.

Exercici 33

33. (dificultat 3) Tres estudiants A, B i C són acusats d'introduir un virus en les sales d'ordinadors de la FIB. Durant l'interrogatori, les declaracions són les següents:

Les tres declaracions són:

- A diu: $b \wedge \neg c$
- B diu: $a \rightarrow c$
- C diu: $\neg c \wedge (a \vee b)$

c) Assumint que ningún va mentir, qui és innocent i qui és culpable?

Si ningú ha mentit, llavors B és culpable.

Exercici 34

34. (dificultat 2) Inventa i defineix formalment alguna altre lògica diferent a la lògica proposicional. Per exemple, si les interpretacions són funcions $I : \mathcal{P} \to \{0, 1, \perp\}$ que també poden donar "indefinit" $\perp$, es pot adaptar la noció de satisfacció de manera raonable, tot i que la resposta ja no serà binaria: l'"avaluació" d'una fórmula F en una interpretació I pot donar 1 (I satisfà F) o 0 (I no satisfà F) o $\perp$ (indefinit).

Podem fer-ho així, de manera raonable, suposant que $\perp$ en la nostra aplicació modela "no ho sé":

```
if ( x and y ) {
   ...
}
```

Exercici 34

34. (dificultat 2) Inventa i defineix formalment alguna altre lògica diferent a la lògica proposicional. Per exemple, si les interpretacions són funcions $I : \mathcal{P} \rightarrow \{0, 1, \perp\}$ que també poden donar "indefinit" $\perp$, es pot adaptar la noció de satisfacció de manera raonable, tot i que la resposta ja no serà binaria: l'"avaluació" d'una fórmula F en una interpretació I pot donar 1 (I satisfà F) o 0 (I no satisfà F) o $\perp$ (indefinit).

$eval_I(\neg F) =$

0	dona	1
1		0
$\perp$		$\perp$

$eval_I(F \wedge G) =$

0	0	dona	0
0	1		0
0	$\perp$		0
1	0		0
1	1		1
1	$\perp$		$\perp$
$\perp$	0		0
$\perp$	1		$\perp$
$\perp$	$\perp$		$\perp$

$eval_I(F \vee G) =$

0	0	dona	0
0	1		1
0	$\perp$		$\perp$
1	0		1
1	1		1
1	$\perp$		1
$\perp$	0		$\perp$
$\perp$	1		1
$\perp$	$\perp$		$\perp$

Exercici 34

34. (dificultat 2) Inventa i defineix formalment alguna altre lògica diferent a la lògica proposicional. Per exemple, si les interpretacions són funcions $I : \mathcal{P} \to \{0, 1, \perp\}$ que també poden donar "indefinit" $\perp$, es pot adaptar la noció de satisfacció de manera raonable, tot i que la resposta ja no serà binaria: l' "avaluació" d'una fórmula F en una interpretació I pot donar 1 (I satisfà F) o 0 (I no satisfà F) o $\perp$ (indefinit).

I si $\perp$ modela "no termina"? Per exemple, en un programa com:

```
if ( not f(...) ) {
}

if ( f(...) and g(...) ) {
}

if ( f(...) or g(...) ) {
}
```

Exercici 34

34. (dificultat 2) Inventa i defineix formalment alguna altre lògica diferent a la lògica proposicional. Per exemple, si les interpretacions són funcions $I : \mathcal{P} \rightarrow \{0, 1, \perp\}$ que també poden donar "indefinit" $\perp$, es pot adaptar la noció de satisfacció de manera raonable, tot i que la resposta ja no serà binaria: l'"avaluació" d'una fórmula F en una interpretació I pot donar 1 (I satisfà F) o 0 (I no satisfà F) o $\perp$ (indefinit).

$eval_I(\neg F) =$

0	dona	1
1		0
$\perp$		$\perp$

$eval_I(F \wedge G) =$

0	0	dona	0
0	1		0
0	$\perp$		0
1	0		0
1	1		1
1	$\perp$		$\perp$
$\perp$	0		$\perp$
$\perp$	1		$\perp$
$\perp$	$\perp$		$\perp$

$eval_I(F \vee G) =$

0	0	dona	0
0	1		1
0	$\perp$		$\perp$
1	0		1
1	1		1
1	$\perp$		1
$\perp$	0		$\perp$
$\perp$	1		$\perp$
$\perp$	$\perp$		$\perp$

Exercici 35

35. (dificultat 2) Com l'exercici anterior, però considerant $I : \mathcal{P} \rightarrow [0 \ldots 1]$, és a dir, l'interpretació d'un símbol p és una probabilitat (un número real entre 0 i 1). En aquest cas, l'avaluació d'una fórmula F en una interpretació I pot donar quelcom (remotamente) semblant a la probabilitat de satisfacció de F en I. En la lògica que has definit, l'avaluació de F en una I determinada, i la de $F \land F$ en aquesta mateixa I donen el mateix resultat?

$$
\begin{aligned}
eval_I(\neg F) &= 1 - eval_I(F) \\
eval_I(F \land G) &= eval_I(F) \cdot eval_I(G) \\
eval_I(F \lor G) &= (eval_I(F) + eval_I(G)) - (eval_I(F) \cdot eval_I(G))
\end{aligned}
$$

Això ens ho hem inventat, però és incorrecte en general perquè les probabilitats de les subfórmules no són independents. Per exemple, l'avaluació de $F \land F$ hauria de donar el mateix que la de F i aquí no és així.

Temari

Deducció en Lògica Proposicional

Capítol 3 dels apunts: ☞ `p3.pdf`

1. Formes normals i clàusules
2. Exercici 1 [construir CNF, DNF]
3. Exercici 2 [construir CNF]
4. Exercici 3 [circuits lògics]
5. Exercici 4 [conjunt de clàusules]
6. Exercici 5 [forma clausal]
7. Exercici 6 [forma clausal]

Formes normals i clàusules

Col·lecció d'apunts bàsics de lògica. ☞ Fitxer `p3.pdf`

- Fórmules com a conjunts
- Literals
- CNF i DNF
- Clàusula
- Conjunt de clàusules
- Clàusula buida
- Clàusula de Horn

- **Fórmules com a conjunts**

$$\begin{array}{lcll} F \wedge F & \equiv & \mathrm{F} & \text{idempotència de } \wedge \\ F \wedge G & \equiv & G \wedge F & \text{conmutativitat de } \wedge \\ (F \wedge G) \wedge H & \equiv & F \wedge (G \wedge H) & \text{associativitat de } \wedge \\ & + & & \end{array}$$

Lema de Substitució

- Conjunció de fórmules. Exemple:

$$\begin{aligned} F &= (F_1 \wedge ((F_2 \wedge (F_3 \wedge F_4) \wedge F_2))) \\ &\equiv F_1 \wedge F_2 \wedge F_3 \wedge F_4 \\ F &= \{\, F_1,\ F_2,\ F_3,\ F_4 \,\} \end{aligned}$$

- Idem amb la connectiva $\vee$

Formes normals i clàusules

- **Literals**

  literals positius: $p, q, r \ldots$

  literals negatius: $\neg p, \neg q, \ldots$

- **CNF** (*conjuntive normal form*)

  conjunció de disjuncions de literals:

  $(l_{1,1} \vee \ldots \vee l_{1,k_1}) \wedge \ \ldots \ \wedge (l_{n,1} \vee \ldots \vee l_{n,k_n})$

- **DNF** (*disjuntive normal form*)

  disjunció de conjuncions de literals:

  $(l_{1,1} \wedge \ldots \wedge l_{1,k_1}) \vee \ \ldots \ \vee (l_{n,1} \wedge \ldots \wedge l_{n,k_n})$

- **Clàusula**

  disjunció de literals:

  $l_1 \vee \ldots \vee l_k \qquad \{l_1, \ldots, l_k\}$

Formes normals i clàusules

- **Literals**

  literals positius: $p, q, r \ldots$

  literals negatius: $\neg p, \neg q, \ldots$

- **CNF** (*conjuntive normal form*)

  conjunció de disjuncions de literals:

  $(l_{1,1} \vee \ldots \vee l_{1,k_1}) \wedge \ldots \wedge (l_{n,1} \vee \ldots \vee l_{n,k_n})$

- **DNF** (*disjuntive normal form*)

  disjunció de conjuncions de literals:

  $(l_{1,1} \wedge \ldots \wedge l_{1,k_1}) \vee \ldots \vee (l_{n,1} \wedge \ldots \wedge l_{n,k_n})$

- **Clàusula**

  disjunció de literals:

  $l_1 \vee \ldots \vee l_k$

  $p_1 \vee \ldots \vee p_m \vee \neg q_1 \vee \ldots \vee \neg q_n$

- CNF $=$ Conjunció de clàusules $=$ **Conjunt de clàusules**

Formes normals i clàusules

- **Clàusula buida**
  - Disjunció de 0 literals: $l_1 \vee \ldots \vee l_k$ amb $k = 0$.
    Es denota amb $\Box$ En fitxer de text: []
  - No és una fórmula segons la sintaxi de la LProp ...
- Extensió de la *sintaxi* de la LProp
  Si tenim que $n \geq 0$ i que $F_1, \ldots, F_n$ són fórmules,
  llavors també són fórmules:
  $\bigwedge_{i \in 1..n} F_i$ i $\bigvee_{i \in 1..n} F_i$
- Extensió de la *semàntica* de la LProp.
  Si I és una interpretació:

$$eval_I(\bigwedge_{i \in 1..n} F_i) = min(\ 1,\ eval_I(F_1),\ \ldots,\ eval_I(F_n)\ )$$
$$eval_I(\bigvee_{i \in 1..n} F_i) = max(0,\ eval_I(F_1),\ \ldots,\ eval_I(F_n)\ )$$

☞ Si $n = 0$ la conjunció de 0 fórmules és trivialment certa

☞ Si $n = 0$ la disjunció de 0 fórmules és trivialment falsa

Formes normals i clàusules

- **Clàusula de Horn**

  $p_1 \vee \ldots \vee p_m \vee \neg q_1 \vee \ldots \vee \neg q_n \quad \text{amb } m \leq 1$

  ☞ màxim 1 literal positiu

Exercici 1

1. (dificultat 2) Demostra que, per a tota fórmula F, hi ha com a mínim una fórmula lògicamente equivalent que està en DNF. Ídem per a CNF. Ajuda: obtenir les fórmulas en CNF i DNF a partir de la taula de veritat per a F.

	p	q	r	F
v_0	0	0	0	0
v_1	0	0	1	1
v_2	0	1	0	0
v_3	0	1	1	0
v_4	1	0	0	1
v_5	1	0	1	0
v_6	1	1	0	1
v_7	1	1	1	0

$$\begin{aligned}\text{DNF: } F' &= v_1 \vee v_4 \vee v_6 \\ &= (\neg p \wedge \neg q \wedge r) \quad \vee \\ &\ (p \wedge \neg q \wedge \neg r) \quad \vee \\ &\ (p \wedge q \wedge \neg r)\end{aligned}$$

Exercici 1

1. (dificultat 2) Demostra que, per a tota fórmula F, hi ha com a mínim una fórmula lògicamente equivalent que està en DNF. Ídem per a CNF. Ajuda: obtenir les fórmulas en CNF i DNF a partir de la taula de veritat per a F.

	p	q	r	F
v_0	0	0	0	0
v_1	0	0	1	1
v_2	0	1	0	0
v_3	0	1	1	0
v_4	1	0	0	1
v_5	1	0	1	0
v_6	1	1	0	1
v_7	1	1	1	0

$$
\begin{aligned}
\text{CNF: } F'' &= \neg v_0 \wedge \neg v_2 \wedge \neg v_3 \wedge \neg v_5 \wedge \neg v_7 \\
&= \neg(\neg p \wedge \neg q \wedge \neg r) \quad \wedge \\
&\quad\ \neg(\neg p \wedge q \wedge \neg r) \quad \wedge \\
&\quad\ \neg(\neg p \wedge q \wedge r) \quad \wedge \\
&\quad\ \neg(p \wedge \neg q \wedge r) \quad \wedge \\
&\quad\ \neg(p \wedge q \wedge r)
\end{aligned}
$$

1. (dificultat 2) Demostra que, per a tota fórmula F, hi ha com a mínim una fórmula lògicamente equivalent que està en DNF. Ídem per a CNF. Ajuda: obtenir les fórmulas en CNF i DNF a partir de la taula de veritat per a F.

	p	q	r	F
v_0	0	0	0	0
v_1	0	0	1	1
v_2	0	1	0	0
v_3	0	1	1	0
v_4	1	0	0	1
v_5	1	0	1	0
v_6	1	1	0	1
v_7	1	1	1	0

$$
\begin{aligned}
\text{CNF: } F'' &= \neg v_0 \wedge \neg v_2 \wedge \neg v_3 \wedge \neg v_5 \wedge \neg v_7 \\
&= (p \vee q \vee r) \wedge \\
&\quad (p \vee \neg q \vee r) \wedge \\
&\quad (p \vee \neg q \vee \neg r) \wedge \\
&\quad (\neg p \vee q \vee \neg r) \wedge \\
&\quad (\neg p \vee \neg q \vee \neg r)
\end{aligned}
$$

Exercici 2

2. (dificultat 3) Dona una manera de calcular una fórmula $\hat{F}$ en CNF para una fórmula F donada (amb $\hat{F} \equiv F$) sense necessitat de construir prèviament la taula de veritat. Ajuda: aplica equivalències lògiques com les lleis de De Morgan i la distributivitat i el Lema de Substitució.

1. Moure les negacions cap a dins:

$\neg\neg F$	$\Longrightarrow$	F	doble negació
$\neg(F \wedge G)$	$\Longrightarrow$	$\neg F \vee \neg G$	llei de De Morgan 1
$\neg(F \vee G)$	$\Longrightarrow$	$\neg F \wedge \neg G$	llei de De Morgan 2

2. Aplicar distributivitat 1:
   $(F \wedge G) \vee H \quad \Longrightarrow \quad (F \vee H) \wedge (G \vee H)$
3. Eliminar literals repetits en les clàusules, i clàusules trivialment certes

Exercici 3

3. (dificultat 3) Cada fórmula de lògica proposicional pot veure's com un circuit electrònic que té una porta lògica per cada connectiva $\wedge$, $\vee$, $\neg$ que aparegui en la fórmula (encara que les fórmules tenen estructura d'arbre, mentre els circuits en realitat permeten compartir subarbres repetits, és a dir, són grafs dirigits acíclics).

El problema del *disseny lògic* consisteix a trobar un circuit adequat que implementi una funció booleana donada. Per a aconseguir circuits *ràpids*, ens va bé representar la funció booleana com una fórmula en CNF (o DNF), perquè la *profunditat* del circuit serà com a màxim tres.

Però també és important utilitzar el mínim nombre de connectives (portes lògiques). Els mètodes d'obtenir CNFs vists en els exercicis anteriors, ens donen la CNF més curta en aquest sentit? Se t'acut alguna millora?

Exercici 4

4. (dificultat 1) La clàusula buida $\Box$ és el cas més senzill de fórmula insatisfactible. Una CNF que sigui un conjunt de zero clàusules, és satisfactible o insatisfactible?

Una CNF és una conjunció de clàusules.
Si es tracta d'una conjunció de zero clàusules, per definició és trivialment certa, i per tant, satisfactible.

Exercici 5

5. (dificultat 2) Demostra que una clàusula és una tautologia si, i només si, conté alhora p i $\neg p$ per un cert símbol proposicional p.

Demostrarem que una clàusula de la forma $p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n$ NO és tautologia ssi NO conté alhora p i $\neg p$.

$p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n$ NO és tautologia ssi

$\exists\, I$, tal que I no és model de $p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n$ ssi

$\exists\, I$, tal que $I \not\models p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n$ ssi

$\exists\, I$, tal que $eval_I(\, p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n\,) = 0$ ssi

$\exists\, I$, tal que $max(eval_I(p_1), \ldots, eval_I(p_m), eval_I(\neg q_1), \ldots, eval_I(\neg q_n)) = 0$ ssi

$\exists\, I$, tal que $I(p_i) = 0$ per a totes les p_i i $I(q_j) = 1$ per a totes les q_j ssi

$p_i \neq q_j$ per a tota i, j

Exercici 6

6. (dificultat 2) Sigui S un conjunt de clàusules amb $\Box \notin S$. Demostra que S és satisfactible (donant un model per a S) en cadascuna de les següents situacions:

a) Tota clàusula C de S té algún literal positiu.

b) Tota clàusula C de S té algún literal negatiu.

c) Per tot símbol de predicat p es compleix que: o bé p apareix només en literals positius en S, o bé p apareix només en literals negatius en S.

Exercici 6

6. (dificultat 2) Sigui S un conjunt de clàusules amb $\square \notin S$.
Demostra que S és satisfactible (donant un model per a S) en cadascuna de les següents situacions:

a) Tota clàusula C de S té algún literal positiu.

C és de la forma: $p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n \quad (m > 1, n \geq 0)$

Sigui $S = \{C_1, C_2, \ldots\}$. Cada clàusula C_i te almenys un literal positiu (un símbol de predicat sense negar).

Un model que satisfarà totes les clàusules de S és la I tal que $I(p) = 1$ per tot símbol de predicat p.

Exercici 6

6. (dificultat 2) Sigui S un conjunt de clàusules amb $\Box \notin S$. Demostra que S és satisfactible (donant un model per a S) en cadascuna de les següents situacions:

b) Tota clàusula C de S té algún literal negatiu.

Sigui $S = \{C_1, C_2, \ldots\}$. Cada clàusula C_i te almenys un literal negatiu (un símbol de predicat negat).

Un model que satisfarà totes les clàusules de S és la I tal que $I(p) = 0$ per tot símbol de predicat p.

Exercici 6

6. (dificultat 2) Sigui S un conjunt de clàusules amb $\Box \notin S$.
Demostra que S és satisfactible (donant un model per a S) en cadascuna de les següents situacions:

c) Per tot símbol de predicat p es compleix que: o bé p apareix només en literals positius en S, o bé p apareix només en literals negatius en S.

És a dir, cada clàusula és de la forma $p_1 \vee \cdots \vee p_m \vee \neg q_1 \vee \cdots \vee \neg q_n$ on

- les p_i's només apareixen en literals positius en la resta de les clàusulas
- les q_i's només apareixen en literals negatius en la resta de les clàusulas

Un model que satisfarà totes les clàusulas de S és la I tal que

$I(p) = 0$ per tot símbol p tal que p només apareix negatiu

$I(p) = 1$ per tot símbol p tal que p només apareix positiu

Exercici 6

6. (dificultat 2) Sigui S un conjunt de clàusules amb $\Box \notin S$. Demostra que S és satisfactible (donant un model per a S) en cadascuna de les següents situacions:

c) Per tot símbol de predicat p es compleix que: o bé p apareix només en literals positius en S, o bé p apareix només en literals negatius en S.

Un model que satisfarà totes les clàusulas de S és la I tal que

$I(p) = 0$ per tot símbol p tal que p només apareix negatiu
$I(p) = 1$ per tot símbol p tal que p només apareix positiu

Nota: en realitat aquesta I satisfarà TOTS els literals de TOTES les clàusules (quan en realitat en tenia prou amb complir UN literal de cada clàusula).

Continguts del capítol 3 [`p3.pdf`] per al proper dia:

- Exercicis del 7 en endavant
- ☞ Resolució (pàg. 5)