

# Lògica en la Informàtica

## Deducció en Lògica Proposicional

José Miguel Rivero   Robert Nieuwenhuis

Facultad de Informàtica  
Universitat Politècnica de Catalunya (UPC)

Tardor 2022



Racó: Col·lecció d'apunts bàsics de lògica. 📄 p3.pf

- Formes normals i clàusules
- Nocions informals de decidibilitat i complexitat
- Resolució. Correcció i completesa
- Resoldre problemes pràctics amb la lògica proposicional

# Deducció en Lògica Proposicional

Necessitem decidir SAT per a fórmules qualssevol, però els SAT solvers només treballen amb CNFs (conjunts de clàusules).

# Deducció en Lògica Proposicional


Necessitem decidir SAT per a fórmules qualssevol, però els SAT solvers només treballen amb CNFs (conjunts de clàusules).

Per tant, necessitem poder transformar fórmules qualssevol en CNFs.

# Dedució en Lògica Proposicional

Necessitem decidir SAT per a fórmules qualssevol, però els SAT solvers només treballen amb CNFs (conjunts de clàusules).

Per tant, necessitem poder transformar fórmules qualssevol en CNFs.

**Tseitin.** Veure la presentació (en la web de Lògica en la Informàtica  <https://www.cs.upc.edu/~rivero/Teaching/LI>) sobre la transformació de Tseitin d'una fórmula qualsevol a una CNF equisatisfactible.

# Deducció en Lògica Proposicional

Per què la transformació via distributivitat pot fer créixer exponencialment la fórmula?

# Deducció en Lògica Proposicional

Per què la transformació via distributivitat pot fer créixer exponencialment la fórmula?

Perquè la regla de distributivitat

$F \vee (G \wedge H) \implies (F \vee G) \wedge (F \vee H)$  DUPLICA la subfórmula  $F$ .

# Dedució en Lògica Proposicional

Per què la transformació via distributivitat pot fer créixer exponencialment la fórmula?

Perquè la regla de distributivitat

$F \vee (G \wedge H) \implies (F \vee G) \wedge (F \vee H)$  DUPLICA la subfórmula  $F$ .

Exemple de cas pitjor: si  $F$  és una DNF  $Cub_1 \vee \dots \vee Cub_n$ , on cada cub és un AND de  $k$  literals, la CNF tindrà TOTES les clàusules possibles amb un literal de cada cub, és a dir  $k^n$  clàusules (el literal del primer cub es pot triar de  $k$  maneres, el del segon també, etc.).



# Dedució en Lògica Proposicional

Per què la transformació via distributivitat pot fer créixer exponencialment la fórmula?

Perquè la regla de distributivitat

$F \vee (G \wedge H) \implies (F \vee G) \wedge (F \vee H)$  DUPLICA la subfórmula  $F$ .

Exemple de cas pitjor: si  $F$  és una DNF  $Cub_1 \vee \dots \vee Cub_n$ , on cada cub és un AND de  $k$  literals, la CNF tindrà TOTES les clàusules possibles amb un literal de cada cub, és a dir  $k^n$  clàusules (el literal del primer cub es pot triar de  $k$  maneres, el del segon també, etc.).

Exemple:  $(p \wedge q \wedge r) \vee (p' \wedge q' \wedge r')$  donaria:

$$\begin{array}{lll} p \vee p', & p \vee q', & p \vee r', \\ q \vee p', & q \vee q', & q \vee r', \\ r \vee p', & r \vee q', & r \vee r' \end{array}$$

Per això fem TSEITIN:

Introduïm un símbol nou per cada connectiva de la fórmula.

I generem les clàusules que "defineixen" el paper que juguen aquests símbols nous en la fórmula.

Per això fem TSEITIN:

Introduïm un símbol nou per cada connectiva de la fórmula.

I generem les clàusules que "defineixen" el paper que juguen aquests símbols nous en la fórmula.

Per exemple, per a expressar que  $p$  és el símbol d'un node OR de dos fills amb símbols  $a$ ,  $b$ , necessitem  $p \Leftrightarrow a \vee b$ .

Per això fem TSEITIN:

Introduïm un símbol nou per cada connectiva de la fórmula.

I generem les clàusules que "defineixen" el paper que juguen aquests símbols nous en la fórmula.

Per exemple, per a expressar que  $p$  és el símbol d'un node OR de dos fills amb símbols  $a$ ,  $b$ , necessitem  $p \Leftrightarrow a \vee b$ .

Per a això:

- expressem  $p \Rightarrow a \vee b$  mitjançant una clàusula de tres literals:  
 $\neg p \vee a \vee b$
- expressem  $p \Leftarrow a \vee b$  que és  $a \Rightarrow p$  i  $b \Rightarrow p$ , amb dues clàusules:  $\neg a \vee p$     $\neg b \vee p$

Per això fem TSEITIN:

Introduïm un símbol nou per cada connectiva de la fórmula.

I generem les clàusules que "defineixen" el paper que juguen aquests símbols nous en la fórmula.

Per a expressar que  $p$  és el símbol d'un node AND de dos fills amb símbols  $a$ ,  $b$ , necessitem  $p \Leftrightarrow a \wedge b$ .

Per això fem TSEITIN:

Introduïm un símbol nou per cada connectiva de la fórmula.

I generem les clàusules que "defineixen" el paper que juguen aquests símbols nous en la fórmula.

Per a expressar que  $p$  és el símbol d'un node AND de dos fills amb símbols  $a$ ,  $b$ , necessitem  $p \Leftrightarrow a \wedge b$ .

Per a això:

- expressem  $p \Rightarrow a \wedge b$  que és  $p \Rightarrow a$  i  $p \Rightarrow b$ , amb dues clàusules:  $\neg p \vee a$   $\neg p \vee b$
- expressem  $p \Leftarrow a \wedge b$  mitjançant una clàusula de tres literals:  $\neg a \vee \neg b \vee p$ .

# Deducció en Lògica Proposicional

En la presentació de la web de LI s'introdueixen també símbols i clàusules per als nodes NOT, però això no és necessari.

# Deducció en Lògica Proposicional

En la presentació de la web de LI s'introdueixen també símbols i clàusules per als nodes NOT, però això no és necessari.

Per exemple, per a evitar el primer node NOT i el seu símbol  $e3$ , podem expressar directament que  $e1 \Leftrightarrow e2 \vee \neg e4$ , generant les clàusules:

$$e1 \vee e2 \vee \neg e4,$$

$$\neg e2 \vee e1,$$

$$e4 \vee e1.$$



En la presentació de la web de LI s'introdueixen també símbols i clàusules per als nodes NOT, però això no és necessari.

Per exemple, per a evitar el primer node NOT i el seu símbol  $e3$ , podem expressar directament que  $e1 \Leftrightarrow e2 \vee \neg e4$ , generant les clàusules:

$$\begin{aligned} &e1 \vee e2 \vee \neg e4, \\ &\neg e2 \vee e1, \\ &e4 \vee e1. \end{aligned}$$

## ATENCIÓ: ERRATA

Hi ha un error en la presentació de la web de LI sobre Tseitin: on diu  $e6 \Leftrightarrow q \vee \neg e7$ , ha de dir  $e6 \Leftrightarrow q \vee e7$ .

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $Tseitin(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $Tseitin(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $Tseitin(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel (e1 en l'exemple).

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $\text{Tseitin}(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $\text{Tseitin}(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel (e1 en l'exemple).
2.  $F$  i  $\text{Tseitin}(F)$  són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  $\text{Tseitin}(F)$  és satisfactible

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $\text{Tseitin}(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $\text{Tseitin}(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel ( $e_1$  en l'exemple).
2.  $F$  i  $\text{Tseitin}(F)$  són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  $\text{Tseitin}(F)$  és satisfactible
3.  $F$  i  $\text{Tseitin}(F)$  NO són logicamente equivalents

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $\text{Tseitin}(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $\text{Tseitin}(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel (e1 en l'exemple).
2.  $F$  i  $\text{Tseitin}(F)$  són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  $\text{Tseitin}(F)$  és satisfactible
3.  $F$  i  $\text{Tseitin}(F)$  NO són logicamente equivalents
4. La mida de  $\text{Tseitin}(F)$  és lineal en la mida de  $F$  (3 clàusules per cada connectiva AND o OR de  $F$ ) + l'arrel

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $\text{Tseitin}(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $\text{Tseitin}(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel (e1 en l'exemple).
2.  $F$  i  $\text{Tseitin}(F)$  són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  $\text{Tseitin}(F)$  és satisfactible
3.  $F$  i  $\text{Tseitin}(F)$  NO són logicamente equivalents
4. La mida de  $\text{Tseitin}(F)$  és lineal en la mida de  $F$  (3 clàusules per cada connectiva AND o OR de  $F$ ) + l'arrel
5. Podem obtenir  $\text{Tseitin}(F)$  en temps lineal a partir de  $F$

## Quins resultats obtenim?

Sigui  $F$  una fórmula.

Sigui  $\text{Tseitin}(F)$  la CNF de (el conjunt de les clàusules generades per) la transformació de Tseitin de  $F$ .

Llavors:

1.  $\text{Tseitin}(F)$  té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel (e1 en l'exemple).
2.  $F$  i  $\text{Tseitin}(F)$  són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  $\text{Tseitin}(F)$  és satisfactible
3.  $F$  i  $\text{Tseitin}(F)$  NO són logicamente equivalents
4. La mida de  $\text{Tseitin}(F)$  és lineal en la mida de  $F$  (3 clàusules per cada connectiva AND o OR de  $F$ ) + l'arrel
5. Podem obtenir  $\text{Tseitin}(F)$  en temps lineal a partir de  $F$
6. Podem reconstruir fàcilment un model de  $F$  a partir d'un model de  $\text{Tseitin}(F)$  ("oblidant-nos" dels símbols auxiliars)



# Dedució en Lògica Proposicional

1. Tseitin( $F$ ) té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel ( $e_1$  en l'exemple).
2.  $F$  i Tseitin( $F$ ) són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  
Tseitin( $F$ ) és satisfactible
3.  $F$  i Tseitin( $F$ ) NO són logicament equivalents
4. La mida de Tseitin( $F$ ) és lineal en la mida de  $F$  (3 clàusules per cada connectiva AND o OR de  $F$ ) + l'arrel
5. Podem obtenir Tseitin( $F$ ) en temps lineal a partir de  $F$
6. Podem reconstruir fàcilment un model de  $F$  a partir d'un model de Tseitin( $F$ ) ("oblidant-nos" dels símbols auxiliars)

# Deducció en Lògica Proposicional

1. Tseitin( $F$ ) té clàusules de fins a 3 literals.  
Compte!: hi ha una clàusula unitària (d'1 només literal) que és el símbol auxiliar de l'arrel ( $e_1$  en l'exemple).
2.  $F$  i Tseitin( $F$ ) són EQUISATISFACTIBLES:  $F$  és satisfactible SSI  
Tseitin( $F$ ) és satisfactible
3.  $F$  i Tseitin( $F$ ) NO són logicament equivalents
4. La mida de Tseitin( $F$ ) és lineal en la mida de  $F$  (3 clàusules per cada connectiva AND o OR de  $F$ ) + l'arrel
5. Podem obtenir Tseitin( $F$ ) en temps lineal a partir de  $F$
6. Podem reconstruir fàcilment un model de  $F$  a partir d'un model de Tseitin( $F$ ) ("oblidant-nos" dels símbols auxiliars)

## Nota:

Sabent que SAT per a fórmules  $F$  qualssevol és NP-complet, els punts 1,2,5 impliquen que 3-SAT també és NP-complet.



## Nota:

Si tenim una subfórmula amb ORs (o ANDs) niats, com a  $p \vee (q \vee r)$  podem fer Tseitin com sempre, introduint per cada OR binari un símbol auxiliar i tres clàusules. Però també podem considerar que és una OR de tres entrades  $p \vee q \vee r$ , i generar un sol símbol auxiliar i quatre clàusules per a expressar  $a \Leftrightarrow p \vee q \vee r$ :

$$\neg a \vee p \vee q \vee r$$

$$\neg p \vee a$$

$$\neg q \vee a$$

$$\neg r \vee a$$

Això pot fer-se similarment per a ORs i ANDs de qualsevol nombre d'entrades.

Ara: veure els vídeos de la web de LI sobre:

👉 the Transportation Company

Ara: veure els vídeos de la web de LI sobre:

- 👉 the Transportation Company
- 👉 Codificació de restriccions numèriques en SAT

Ara: veure els vídeos de la web de LI sobre:

- 👉 the Transportation Company
- 👉 Codificació de restriccions numèriques en SAT
  - ALO, AMO, exactly one

Ara: veure els vídeos de la web de LI sobre:

- 👉 the Transportation Company
- 👉 Codificació de restriccions numèriques en SAT
  - ALO, AMO, exactly one
  - cardinality constraints en general:

$$l_1 + \dots + l_n \leq K$$

$$l_1 + \dots + l_n \geq K$$

$$l_1 + \dots + l_n = K$$

Ara: veure els vídeos de la web de LI sobre:

- 👉 the Transportation Company
- 👉 Codificació de restriccions numèriques en SAT

- ALO, AMO, exactly one
- cardinality constraints en general:

$$l_1 + \dots + l_n \leq K$$

$$l_1 + \dots + l_n \geq K$$

$$l_1 + \dots + l_n = K$$

- pseudo-Boolean constraints:

$$a_1 l_1 + \dots + a_n l_n \leq K$$

$$a_1 l_1 + \dots + a_n l_n \geq K$$

$$a_1 l_1 + \dots + a_n l_n = K$$