

Logique des propositions Cours1

Hadjila Fethallah

Maître de Conférences au
Département d'Informatique

F_hadjila@mail.univ-tlemcen.dz



Introduction

- **Logique mathématique**

- discipline qui s'intéresse à l'étude des méthodes et modèles de raisonnement valide

- Elle se divise en 04 branches:

- Théorie de modèles
 - Théorie de preuve
 - Théorie des ensembles
 - calculabilité

Raisons de naissance de L.M

■ Paradoxes (19 & 20 siècles)

- certains domaines mathématiques formalisés comportaient des paradoxes
- géométrie (initialement faite par Euclid), théorie des ensembles (initialement faite par cantor),...

■ **Paradoxe de russell:** soit R l'ensemble des ensembles n'appartenant pas à eux-mêmes, est ce que R appartient-il à lui-même ?

- Si on répond oui, alors, comme par définition les membres de cet ensemble n'appartiennent pas à eux-mêmes, il n'appartient pas à lui-même : contradiction.
- Mais si la réponse est non, alors il a la propriété requise pour appartenir à lui-même : contradiction à nouveau.
- $R = \{X \mid X \notin X\}$, de cette définition on aura: $R \in R \Leftrightarrow R \notin R$

■ Postulat du parallélisme (géométrie)

Paradoxe de Berry

- **Version anglaise**

- "the smallest positive integer not definable in fewer than twelve words"

- **Version française**

- « Le plus petit entier naturel non descriptible par une expression de quinze mots ou moins. »
- Ce nombre appartient-il à l'ensemble des entiers naturels descriptibles par une expression de quinze mots ou moins ?

Raisons de naissance de L.M

- Est-ce que les fondements mathématiques sont cohérents (sans erreurs)
- Programme d'hilbert (1900)
- Une liste de 23 problèmes à résoudre
 - Consistance de l'arithmétique (les propriétés des entiers naturels)
 -
- Théorème d'incomplétude de godel(1931)
 - Toute théorie axiomatique suffisamment expressive(contenant au moins l'arithmétique)est incomplète
 - On ne peut même pas prouver la consistance de l'arithmétique

Définitions

- Proposition (assertion): énoncé déclaratif qui peut être vrai ou faux
- Exemple:
- La terre est ronde
- L'amphi est fermé
- Tous le monde est présent
- $3+4=8$

...

Définitions

- La définition exclut les énoncés impératifs, exclamatifs, et interrogatifs
- Exemples:
 - ☐ Fermes la fenêtre!
 - ☐ Est-ce que la fenêtre est fermée ?
 - ☐ Quelle belle fenêtre!
 - ☐ Il est présent
 - ☐ N est premier
 - ☐ $x+y > 10$
 - ☐ Cette expression est fausse (autoréférence)

Définitions

- Prédicat : relation n-aire définie sur le produit cartésien d'un ensemble de domaines
- Ex: $\text{ami}(X, Y)$, avec $X, Y \in \text{PromotionL2}$
 - $\text{ami} \equiv \{(\text{etu1}, \text{etu3}), (\text{etu2}, \text{etu9}), (\text{etu3}, \text{etu1}), (\text{etu9}, \text{etu2}) \dots\}$.
- Remarque : après le remplacement des variables libres, le prédicat devient une proposition

Logique des propositions

- Le calcul des propositions passe par les étapes suivantes:
 - Comment écrire les formules?
Aspects syntaxiques
 - Comment déterminer la valeur de vérité d'une formule ? Aspects sémantiques
 - Comment démontrer (automatiquement) de nouveaux résultats ? Aspects déductifs

syntaxe

- le vocabulaire (alphabet) de la logique propositionnelle est composé
 - l'ensemble des connecteurs : $\{\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg, (,)\}$
 - l'ensemble des symboles de variables propositionnelles noté $P : \{p, q, r, s, \dots\}$.
- Pour construire une formule correcte (FBF) on applique les règles de grammaire:
- Toute formule $F \in \text{Prop}$ peut être définie comme suit:
- $F \equiv p$, avec p une proposition atomique
- $F \equiv \neg H$, avec $H \in \text{Prop}$
- $F \equiv H \wedge I$, ou $F \equiv H \vee I$, ou $F \equiv H \Rightarrow I$, ou $F \equiv H \Leftrightarrow I$.
avec $H, I \in \text{Prop}$

Syntax

Exemples

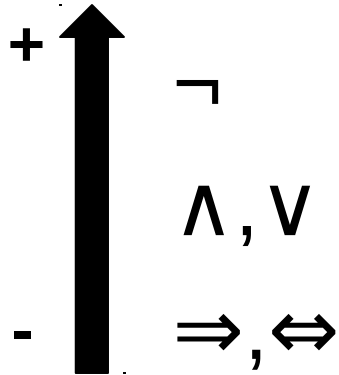
- p
- $(q \Rightarrow p) \Leftrightarrow (s \Rightarrow p)$
- $\neg(p \vee (r \wedge \neg s) \vee t)$
- $\neg\neg p$

Contre exemples:

- $pq \vee$
- $\Rightarrow p$

Syntaxe

■ Priorité:



Sémantique (théorie de modèles)

- La sémantique d'une formule propositionnelle est sa valeur de vérité, ie. 1 ou 0
- On calcule la sémantique grâce à la notion d'interprétation
- Une interprétation I est une paire $(\{0, 1\}, \delta)$, avec δ : une fonction d'interprétation:
- $\delta: \text{Prop} \longrightarrow \{0, 1\}$
- elle affecte à chaque proposition atomique une valeur de vérité, en plus elle représente la sémantique des connecteurs

sémantique

Exemple:

$$F : \neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$$

p	q	$\neg p$	$\neg q$	$p \wedge q$	$\neg(p \wedge q)$	$(\neg p \vee \neg q)$	F
1	1	1	0	1	0	0	1
0	1	1	0	0	1	1	1
1	0	0	1	0	1	1	1
0	0	0	1	0	1	1	1

Chaque ligne de la table de vérité représente une interprétation possible

L'interprétation assurant un 1 est appelée modèle

L'interprétation assurant un 0 est appelée antimodèle

Types de formules propositionnelles

■ Formule satisfaisable (consistante)

- F est satisfaisable ssi $\exists \delta$ tel que: $\delta(F)=1$
- En d'autres termes, Il y a au moins une ligne donnant 1 dans la table de vérité de F

■ exemples

- $(s \Rightarrow p)$
- $(\neg p \vee (r \wedge \neg s))$

Types de formules propositionnelles

■ Formule insatisfaisable (antilogie, inconsistante, contradictoire)

- F est insatisfaisable ssi $\forall \delta : \delta(F)=0$
- En d'autres termes, toutes les lignes de la table de vérité donnent des 0

■ exemples

- $(p \wedge \neg p)$
- $(\neg p \Leftrightarrow p)$

Types de formules propositionnelles

■ Formule valide (tautologie)

- F est valide ssi $\forall \delta : \delta(F)=1$
- En d'autres termes, toutes les lignes de la table de vérité donnent des 1

■ exemples

- $(p \vee \neg p)$
- $(\neg p \Rightarrow \neg s) \Rightarrow (s \Rightarrow p)$

Types de formules propositionnelles

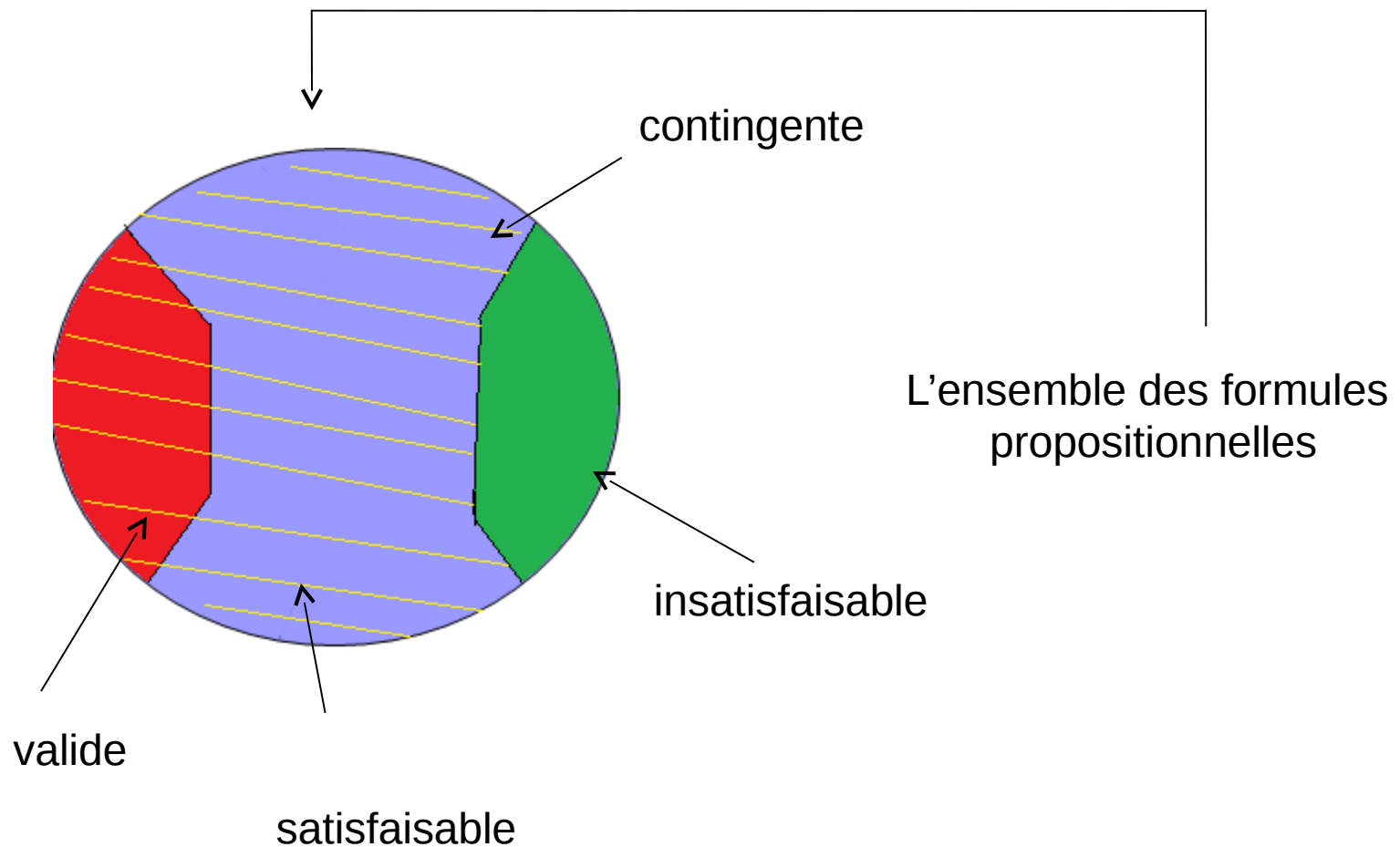
■ Formule contingente

- F est contingente ssi $\exists \delta$ tel que: $\delta(F)=1$, et δ' tel que: $\delta'(F)=0$
- En d'autres termes, Il y a au moins une ligne qui donne 1 et une autre qui donne 0, dans la table de vérité de F

■ exemples

- $(s \Rightarrow p)$
- $(p \vee (r \wedge \neg s))$

Classes de formules



Conséquence logique

- Soit $A=\{F1, \dots, F_n\}$ un ensemble de formules bien formées 'fbf', de même G est une 'fbf'.
- G est une conséquence logique de A ssi:
 - $\forall \delta : \text{si } \delta(F1)=1 \text{ et } \delta(F2)=1 \dots \text{et } \delta(F_n)=1$
Alors $\delta(G)=1$
 - En d'autres termes tous modèle de A est aussi un modèle de G
- On note $A \models G$
 - Exemple: $\{s \Rightarrow p, \neg p\} \models \neg s$
 - N.B: Une conséquence logique de \emptyset est une tautologie

Equivalence logique

- Soit A et G deux formules bien formées 'fbf', A est logiquement équivalente à B ssi:
 - note $A \models G$ et
 - $G \models A$

Formes normales

■ Littéral

- c'est une proposition atomique ou bien sa négation
- Exemples: $\neg p$, q ...

■ Forme Normale Conjonctive

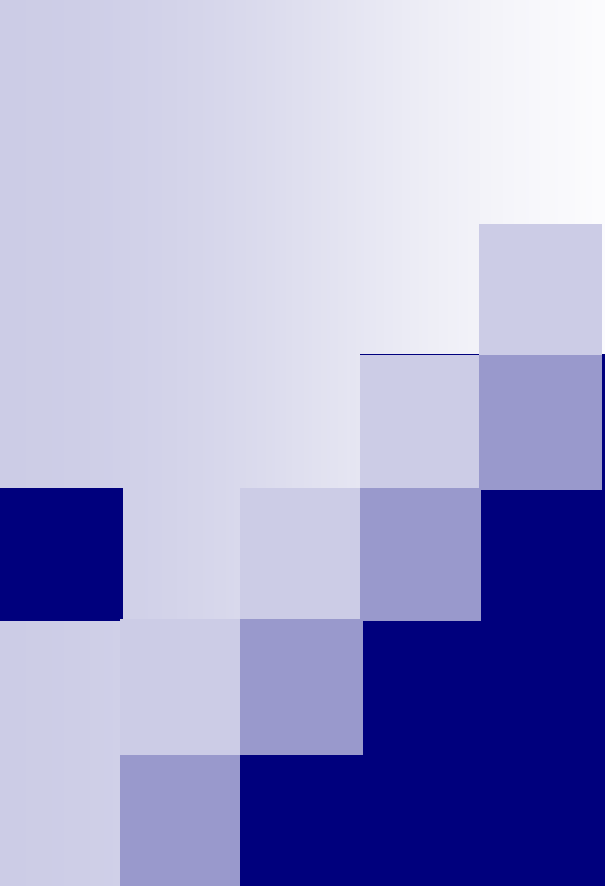
- C'est une conjonction de disjonction de littéraux
- Ex: $(\neg P1 \vee \neg P2 \vee \neg P3) \wedge (P3 \vee \neg P2) \wedge (P1 \vee \neg P3)$

■ Forme Normale disjonctive

- C'est une disjonction de conjonction de littéraux
- Ex: $(\neg P1 \wedge P3) \vee (\neg P3 \wedge \neg P2) \vee (P1 \wedge P2 \wedge P3)$



FIN



Logique des propositions Cours2

Hadjila fethallah

Maître de Conférences au
Département d'Informatique

F_hadjila@mail.univ-tlemcen.dz

Théorie de preuve

- L'approche purement sémantique, basée sur la recherche de modèles n'est pas pratique
- Pour vérifier que $A \models B$, on doit:
 - trouver tous les modèles de A
 - Vérifier si ces modèles sont aussi des modèles pour B
- Si A contient n propositions atomiques alors, il faut parcourir 2^n interprétations (coût exponentiel et non pratique)
- Solution: utiliser une approche syntaxique, ie. On emploie uniquement des règles d'inférences et des axiomes

■ Système formel (système de preuve)

- Un système de preuve est un quadruplet:
 - Un ensemble dénombrable V de symboles
 - Un sous ensemble F de V^* appelé ensemble de formules
 - Un sous ensemble A de F appelé ensemble d'axiomes
 - Un sous ensemble RI de F appelé ensemble de règles d'inférence
- Une règle d'inférence est une implication qui est toujours vraie
- Un axiome est une formule valide
- Exemples
- Axiome : $(p \rightarrow (q \rightarrow p))$
- RI: modus ponens
$$\frac{p, \quad p \rightarrow q}{q}$$

Règles d'inférence

Definition (Modus ponens)

$$p \rightarrow q, p \vdash q$$

Definition (Modus tollens)

$$p \rightarrow q, \neg q \vdash \neg p$$

Definition (Syllogisme)

$$p \rightarrow q, q \rightarrow r \vdash p \rightarrow r$$

démonstration

- Une démonstration dans un système formel S , est une suite d'expressions A_1, \dots, A_n , telle que, tout A_i est soit:
 - Un axiome de S
 - Une conséquence des expressions précédentes, générée avec l'une des règles d'inférence
- Un théorème de S est la dernière expression de la démonstration
- On note: $\vdash A_n$

déductibilité

- Une formule A est déductible de l'ensemble d'hypothèses H ssi, il y a une suite finie d'expressions A_1, \dots, A_n , telle que $A_n = A$, et pour tout $i \in \{1..N\}$, A_i est dans l'une des situations suivantes:
 - A_i est un axiome de S
 - A_i est une conséquence des expressions précédentes, générée avec l'une des règles d'inférence
 - $A_i \in H$
- On note $H \vdash A$

Systemes à la Hilbert

- (logique propositionnelle classique)
- $H2 = (\{ \neg, \rightarrow \}, A1, A2, A3, MP)$
- $A1: (A \rightarrow (B \rightarrow A))$
- $A2: ((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)));$
- $A3: ((\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B))$
- MP (règle d'inférence)
- $(MP) \frac{A, (A \rightarrow B)}{B}$

exemple

- Démontrer $B \rightarrow B$?

Systemes à la Hilbert

- *Logique minimale* = ($\{\neg, \rightarrow\}$, A_1, A_2, A_3 , MP)
- $A_1: (A \rightarrow (B \rightarrow A)) \quad \neg$
- $A_2: ((A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)));$
- $A_3: ((\neg B \rightarrow \neg A) \rightarrow (\neg B \rightarrow A) \rightarrow B)))$
- MP (règle d'inférence)
- $(MP) \frac{A, (A \rightarrow B)}{B}$

Resolution

- Cette règle d'inférence a été inventée par Robinson en 1965.
- Toutes les formules de la resolution respectent la forme normale conjonctive (FNC)
E.g., $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

■ Règle de Resolution unitaire

$$\frac{\ell_1 \vee \dots \vee \ell_k, \quad m}{\ell_1 \vee \dots \vee \ell_{i-1} \vee \ell_{i+1} \vee \dots \vee \ell_k}$$

avec ℓ_i et m sont des littéraux complémentaires :

$$\text{E.g., } \frac{P_{1,3} \vee P_{2,2}, \neg P_{2,2}}{P_{1,3}}$$

Regle de Resolution

Resolution

- On suppose que l_i et m_r sont des litteraux complementaires
- Le resultat est appelé resolvant

$$l_1 \vee \dots \vee l_k, \quad m_1 \vee \dots \vee m_i$$

$$l_1 \vee \dots \vee l_{i-1} \vee l_{i+1} \vee \dots \vee l_k \vee m_1 \vee \dots \vee m_{r-1} \vee m_{r+1} \vee \dots \vee m_i$$

Exemple

- $C1 = (p \vee q \vee \neg r \vee s)$
- $C2 = (q \vee \neg p \vee t)$
- Résolution sur p et $\neg p$.
- Résolvant :
- $(q \vee \neg r \vee s \vee q \vee t),$

Principe de réfutation

- Pour prouver la clause A à partir d'un ensemble de clauses H , il suffit de prouver que H et $\neg A$ sont inconsistantes, c'est-à-dire qu'on peut inférer \square à partir de H et $\neg A$
- Pour démontrer $H \vdash A$, il suffit de démontrer que $\{H, \neg A\} \vdash \square$

Algorithme de résolution

Question :

est ce que $H \vdash A$?

H : un ensemble de formules et A est une formule .

Algorithme

1. H_1 obtenu en remplaçant les formules de H par leur FNC.
2. $H_2 = H_1 \cup \{\neg A\}$. Sachant que $\neg A$ est sous FNC
3. H_3 est obtenu en remplaçant les formules de H_2 par leurs clauses.
4. On teste la règle de résolution pour toute paire (B_j, B_i) où $B_i, B_j \in H_3$, on arrête lorsqu'on obtiendra une formule **insatisfaisable** (\square), dans ce cas: $H \vdash A$ est confirmée
5. Si la clause vide \square ne peut être obtenue alors $H \not\vdash A$

Résolution

Exemple 1 Soient les formules propositionnelles suivantes:

$$H = \left\{ \begin{array}{l} b \wedge c \Rightarrow a \\ b \\ d \wedge e \Rightarrow c \\ e \vee f \\ d \wedge \neg f \end{array} \right\}$$

démontrer a ?

Résolution : Logique propositionnelle

1. Transformation en clauses F.N.C.

$$H1 = \left\{ \begin{array}{l} a \vee \neg b \vee \neg c \\ b \\ c \vee \neg d \vee \neg e \\ e \vee f \\ d \\ \neg f \end{array} \right\}$$

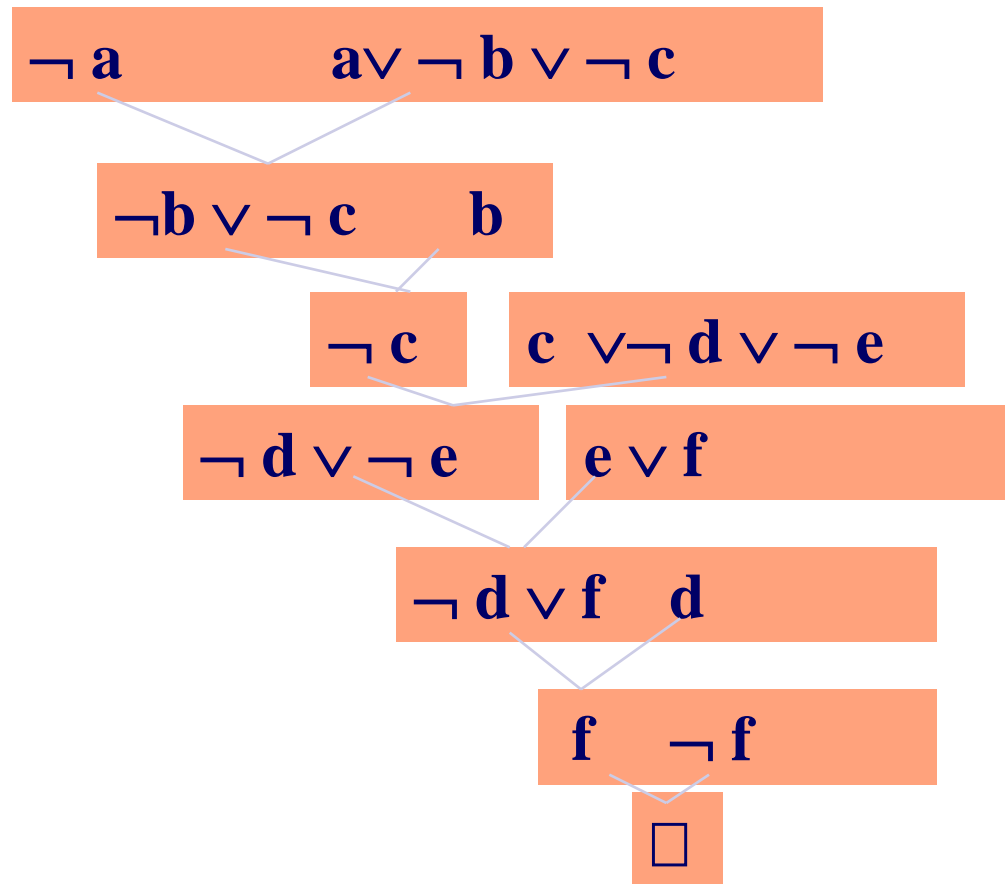
Résolution : Logique propositionnelle

2. Et 3. Ajouter $\neg a$ aux clauses

$H3 = \{$
 $a \vee \neg b \vee \neg c$
 b
 $c \vee \neg d \vee \neg e$
 $e \vee f$
 d
 $\neg f$
 $\neg a$
 $\}$

Résolution

4. Réfutation : Obtenir une contradiction \square (clause vide) à partir de H3



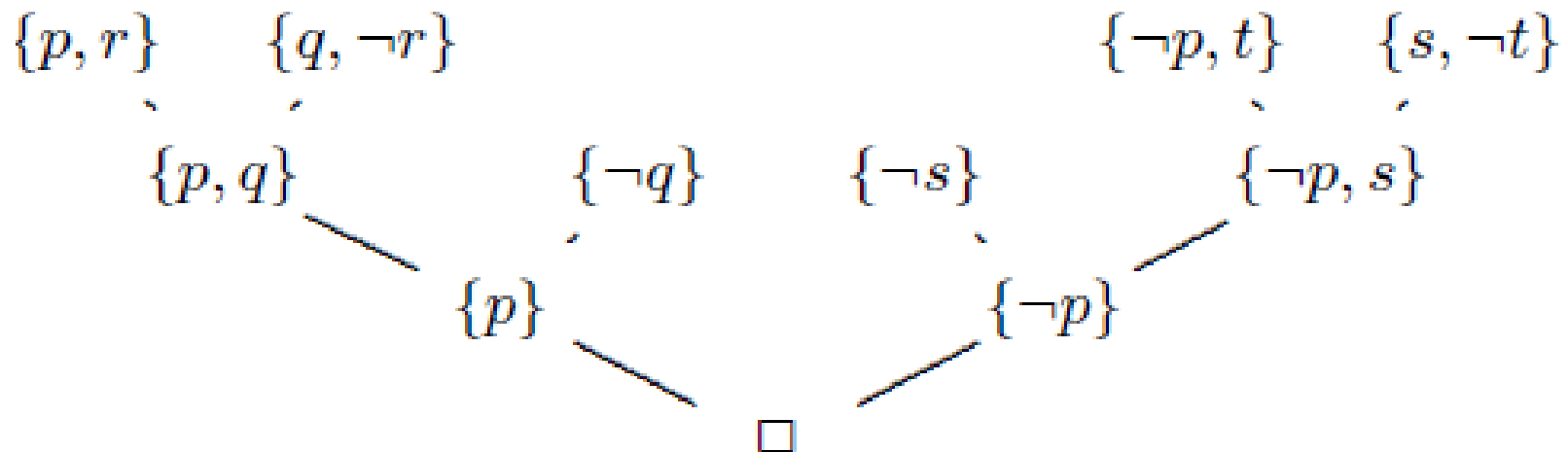
Example 2

$$S \vdash_R \Box ?$$

$$S = (p \vee r) \wedge (q \Leftarrow r) \wedge \neg q \wedge (t \Leftarrow p) \wedge \neg s \wedge (s \Leftarrow t)$$

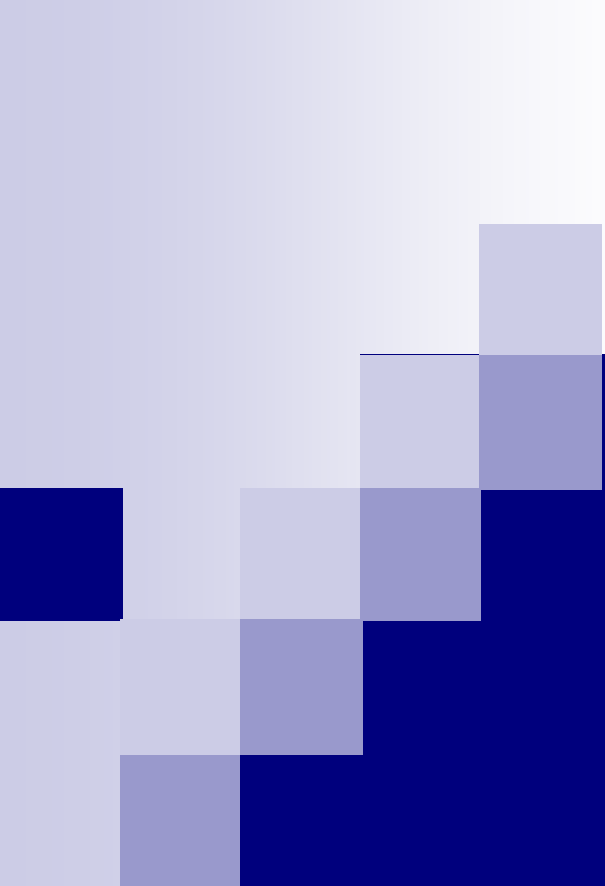
$$S = (p \vee r) \wedge (q \vee \neg r) \wedge \neg q \wedge (\neg p \vee t) \wedge \neg s \wedge (s \vee \neg t)$$

$$S = \{\{p, r\}, \{q, \neg r\}, \{\neg q\}, \{\neg p, t\}, \{\neg s\}, \{s, \neg t\}\}$$





FIN



Logique des propositions Cours3

Hadjila fethallah

Maître de Conférences au
Département d'Informatique

F_hadjila@mail.univ-tlemcen.dz

Calcul de séquents

- L'élément de base de cette approche de démonstration est le séquent.
- Un séquent noté $\Gamma \vdash \Delta$ est constitué de regroupement de propositions.
- Γ et Δ sont des multi ensembles de propositions (une structure de données où l'ordre ne compte pas, mais où les éléments peuvent être répétés), le multi ensemble peut être vide.
- Le séquent suivant $A_1, \dots, A_n \vdash B_1, \dots, B_m$ se lit:
- de A_1 et ...et A_n on peut déduire B_1 ou ...ou B_n .

Définition d'un séquent

Si $\varphi_1, \varphi_2, \dots, \varphi_n, \psi_1, \psi_2, \dots, \psi_m$ sont des singletons alors

le séquent $\varphi_1, \varphi_2, \dots, \varphi_n \vdash \psi_1, \psi_2, \dots, \psi_m$ signifie que:

- ✓ de φ_1, \dots , et de φ_n on peut déduire ψ_1 ou ψ_2, \dots , ou ψ_m
- ✓ Ou $(\varphi_1 \wedge \varphi_2 \wedge \dots \wedge \varphi_n) \Rightarrow (\psi_1 \vee \psi_2 \vee \dots \vee \psi_m)$

Ce système de preuve se compose

- ✓ D'un seul axiome
- ✓ 15 règles d'inférences

Calcul des séquents : Règles

Les règles d'inférence du calcul des séquents se repartissent en 03 catégories:



Règles structurelles



Règles logiques



Règles de coupure

Axiomes

L'axiome du calcul des séquents est de la forme:

$$\Gamma_1, \varphi, \Gamma_n \vdash \Delta_1, \varphi, \Delta_m$$

Intuition:

à partir d'un certain nombre de formules, incluant φ , on peut dériver entre autres

φ

Règles structurelles

Les règles structurelles permettent de négliger l'ordre ou la redondance des formules dans un séquent.

1. Règle de permutation dans l'antécédent, notée G_P (Permutation Gauche) :

$$\frac{\Gamma_1, \varphi, \psi, \Gamma_n \vdash \Delta}{\Gamma_1, \psi, \varphi, \Gamma_n \vdash \Delta} \quad G_P$$

2. Règle de permutation dans la conclusion, notée D_P (Permutation Droite) :

$$\frac{\Gamma \vdash \Delta_1, \varphi, \psi, \Delta_2}{\Gamma \vdash \Delta_1, \psi, \varphi, \Delta_2} \quad D_P$$

Règles structurelles (...)

3. Règle de contraction dans l'antécédent, notée G_C (Contraction, Gauche) :

$$\frac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi \vdash \Delta} \quad G_C$$

4. Règle de contraction dans la conclusion, notée D_C (Contraction Droite) :

$$\frac{\Gamma \vdash \Delta, \varphi, \varphi}{\Gamma \vdash \Delta, \varphi} \quad D_C$$

Règles structurelles (...)

5. Règle d'atténuation (ou d'affaiblissement) dans la condition, notée G_A
(Atténuation Gauche) :

$$\frac{\Gamma \vdash \Delta}{\Gamma, \varphi \vdash \Delta} \quad G_A$$

6. Règle d'atténuation (ou d'affaiblissement) dans la conclusion, notée G_A
(Atténuation Droite) :

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, \varphi} \quad D_A$$

Règles Logiques



Passage des prémisses à la conclusion

1. Règle d'introduction de la négation dans la condition, notée $G \neg$

$$\frac{\Gamma \vdash \varphi, \Delta}{\Gamma, \neg\varphi \vdash \Delta} \quad G \neg$$

Intuition:

si de Γ on peut dériver φ ou Δ , alors si φ est fausse, Δ est nécessairement vrai.

Règles Logiques (2)

2. Règle d'introduction de la négation dans la conclusion, notée D_{\neg}

$$\frac{\Gamma, \varphi \vdash \Delta}{\Gamma \vdash \neg \varphi, \Delta} \quad D_{\neg}$$

Intuition:

Si de Γ et φ on peut dériver Δ , alors de Γ seul, on peut inférer soit $\neg \varphi$ soit

Δ . En réalité, si on suppose que $\neg \varphi$ et Δ sont faux en même temps, donc φ serait vrai, on aura par conséquent Γ et φ vraies, mais Δ faux. ceci contredit le séquent de la condition.

Règles Logiques (3)

3. Règle d'introduction de la conjonction dans l'antécédent, notée G_{\wedge}

$$\frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta} \quad G_{\wedge}$$

Intuition:

Par convention, la condition $\varphi \wedge \psi$ est notée comme suit: φ, ψ

Règles Logiques (4)

4. Règle d'introduction de la conjonction dans la conclusion, notée D_{\wedge}

$$\frac{\Gamma \vdash \varphi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \varphi \wedge \psi, \Delta} \quad D_{\wedge}$$

Règles Logiques (5)

5. Règle d'introduction de la disjonction dans l'antécédent, notée $G \vee$

$$\frac{\Gamma, \varphi \vdash \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta} \quad G \vee$$

Règles Logiques (6)

6. Règle d'introduction de la disjonction dans la conclusion, notée $D \vee$

$$\frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta} \quad D \vee$$

Intuition:

Par convention, la conclusion $\varphi \vee \psi$ est notée comme suit: φ, ψ

Règles Logiques (7)

7. Règle d'introduction de l'implication dans l'antécédent, notée $G \Rightarrow$

$$\frac{\Gamma_1, \Gamma_2 \vdash \Delta_1, \varphi, \Delta_2 \quad \Gamma_1, \psi, \Gamma_2 \vdash \Delta_1, \Delta_2}{\Gamma_1, \varphi \Rightarrow \psi, \Gamma_2 \vdash \Delta_1, \Delta_2} \quad G \Rightarrow$$

Règles Logiques (8)

8. Règle d'introduction du conditionnel dans la conclusion, notée $D \Rightarrow$

$$\frac{\Gamma, \varphi \vdash \psi, \Delta}{\Gamma \vdash \varphi \Rightarrow \psi, \Delta} \quad D \Rightarrow$$

Règles de coupure

1. Règle court-circuitant les résultats intermédiaires notée C

$$\frac{\Gamma_1 \vdash \varphi, \Delta_1 \quad \Gamma_2, \varphi \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2} \text{ C}$$

Intuition:

Si de Γ_1 , on peut dériver φ ou Δ_1 , on peut en faire la même chose en partant de Γ_1, Γ_2 (en appliquant la règle d'atténuation).

En plus, si de Γ_2, φ , on peut dériver Δ_2 on peut faire la même chose en partant de Γ_1, Γ_2 et φ . (en appliquant la règle d'atténuation).

ainsi, on peut dériver Δ_1 ou Δ_2 , à partir de Γ_1, Γ_2

Exemples

Soit le séquent

$$\vdash \varphi \vee \neg \varphi$$

$\frac{}{\varphi \vdash \varphi}$	Ax
$\frac{\varphi \vdash \varphi}{\vdash \neg \varphi, \varphi}$	$D \neg$
$\frac{\vdash \neg \varphi, \varphi}{\vdash \varphi, \neg \varphi}$	$D p$
$\frac{\vdash \varphi, \neg \varphi}{\vdash \varphi \vee \neg \varphi}$	$D \vee$

Exemples

Soit le séquent

$$\vdash \varphi \Rightarrow (\psi \Rightarrow (\varphi \wedge \psi))$$

$$\begin{array}{c} \frac{}{\varphi, \psi \vdash \varphi} Ax \quad \frac{}{\varphi, \psi \vdash \psi} Ax \\ \hline \varphi, \psi \vdash \varphi \wedge \psi \quad D \wedge \\ \hline \varphi \vdash \psi \Rightarrow (\varphi \wedge \psi) \quad D \Rightarrow \\ \hline \vdash \varphi \Rightarrow (\psi \Rightarrow (\varphi \wedge \psi)) \quad D \Rightarrow \end{array}$$

propriétés du calcul de séquents

 Le séquent $\Gamma \vdash \Delta$ est alors dit valide si $I(\Gamma \vdash \Delta) = 1$ pour toute interprétation I

Notation: $\Gamma \vdash \Delta$

 propriété de correction (soundness)

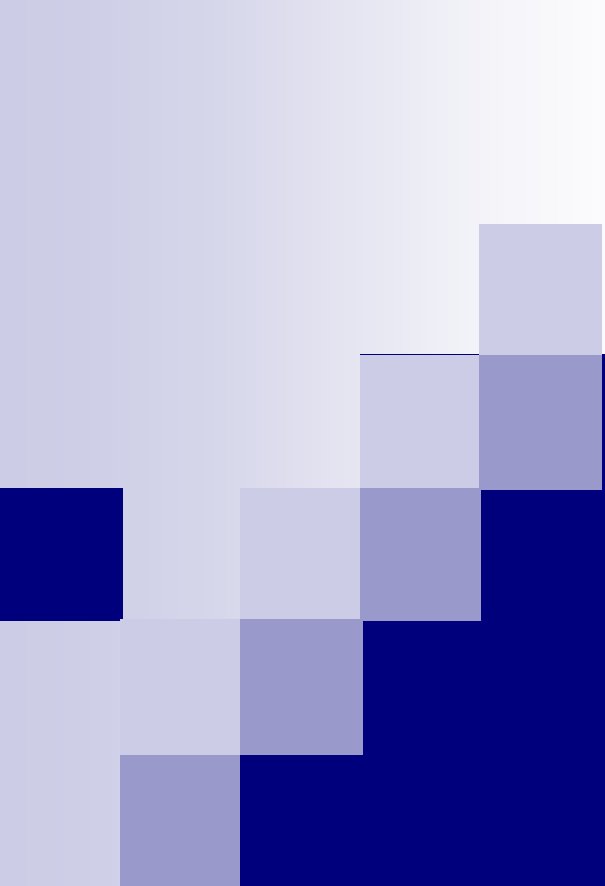
S'il existe une preuve de $\Gamma \vdash \Delta$ dans le calcul des séquents, alors $\Gamma \vdash \Delta$

 propriété de complétude (completeness)

Si $\Gamma \vdash \Delta$, alors il existe une preuve de $\Gamma \vdash \Delta$ dans le calcul des séquents.



FIN



Logique des propositions Cours4

Hadjila fethallah

Maître de Conférences au
Département d'Informatique

F_hadjila@mail.univ-tlemcen.dz

Motivation des solveurs SAT

- La resolution n'est plus efficace Lorsque la formule FNC est satisfaisable
- Meme si la formule est insatisfaisable, la recherche risque d'etre longue à cause du manque d'orientation
- Besoin de nouveaux algorithmes !

Satisfaisabilité propositionnelle (problèmes SAT)

- ✓ 02 grandes familles pour vérifier la satisfaisabilité propositionnelle:
- ✓ Les algorithmes complets (avec back tracking)
 - algorithme DPLL (Davis, Putnam, Logemann, Loveland, 62)
- ✓ algorithmes incomplets (recherche locale)
 - Algorithme WalkSAT
- ✓ Les problèmes sont faciles si la taille de chaque clause ≤ 2
- ✓ Généralement difficiles pour une taille ≥ 3

Algorithme DPLL

Determine si une formule en format FNC est satisfaisable ou non

Amelioration par rapport à la table de verité:

1. Arrêt précoce

une clause est vraie, si au moins un littéral est vrai.

La formule est fausse, si au moins une clause est fausse.

2. Heuristique du symbole pur (monotone)

Un symbole est pur: s'il apparait toujours avec le même "signe" dans toutes les clauses.

Ex: $(A \vee \neg B)$, $(\neg B \vee \neg C)$, $(C \vee A)$, A , B sont purs, C ne l'est pas.

On affecte le symbole pur à vrai

3. Heuristique de la clause unitaire

Clause unitaire: un seul littéral dans la clause

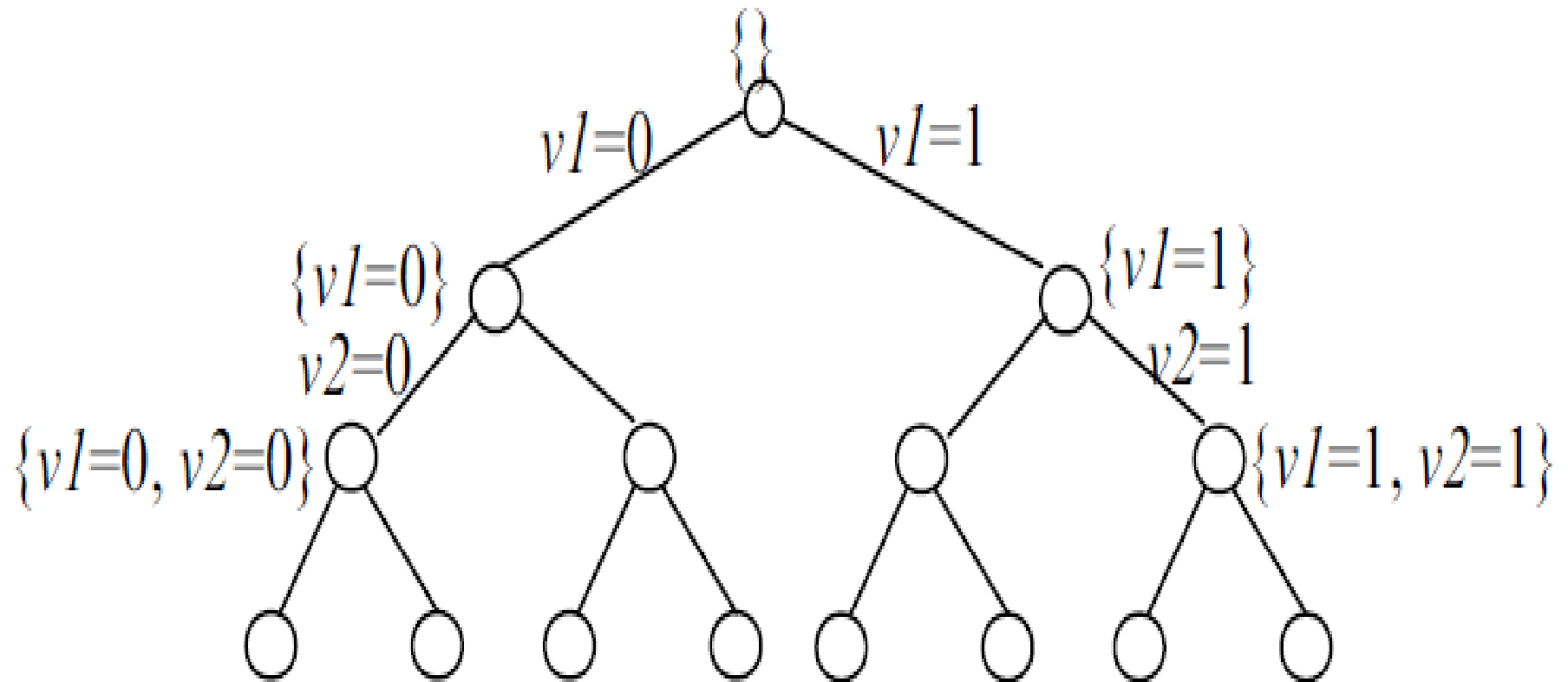
On affecte ce littéral à vrai

.

Algorithme DPLL

- **fonction** DPLL(Φ) retourne Vrai ou Faux
- **si** Φ ne contient aucune clause **alors retourner** *vrai*;
- **si** Φ contient une clause vide **alors retourner** *faux*;
- **tant qu'il existe une** clause unitaire L **dans** Φ
 - $\Phi \leftarrow \text{propagation-unitaire}(L, \Phi)$;
- **tant qu'il existe un** littéral-pur L **dans** Φ
 - $\Phi \leftarrow \text{affecter-littéral-pur}(L, \Phi)$;
- $I \leftarrow \text{choisir-littéral}(\Phi)$;
- **Si** DPLL($\Phi_{\neg I}$)=*vrai* **alors retourner** *vrai*
sinon retourner DPLL($\Phi_{\neg \neg I}$)

Arbre de recherche



satisfaisabilité

- Simuler DPLL sur les 3-FNC suivantes:

- ✓ $(\neg D \vee \neg B \vee C) \wedge (B \vee \neg A \vee \neg C) \wedge (\neg C \vee \neg B \vee E) \wedge (E \vee \neg D \vee B) \wedge (B \vee E \vee \neg C)$

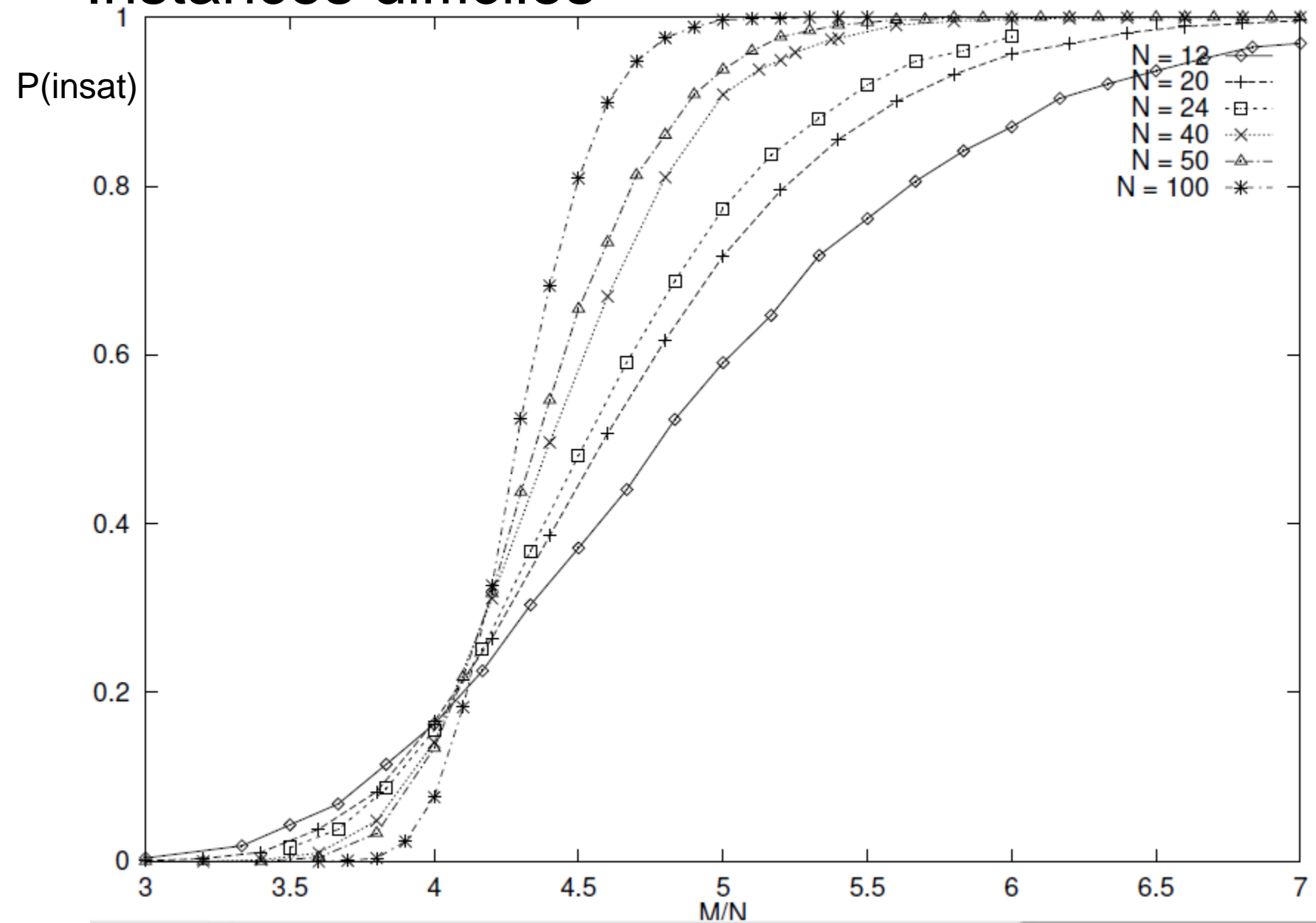
- ✓ $(\neg A \vee \neg B \vee \neg C) \wedge (\neg A \vee B) \wedge (\neg B \vee C) \wedge (\neg C \vee A) \wedge (A \vee B \vee C)$

M = nombre de clauses

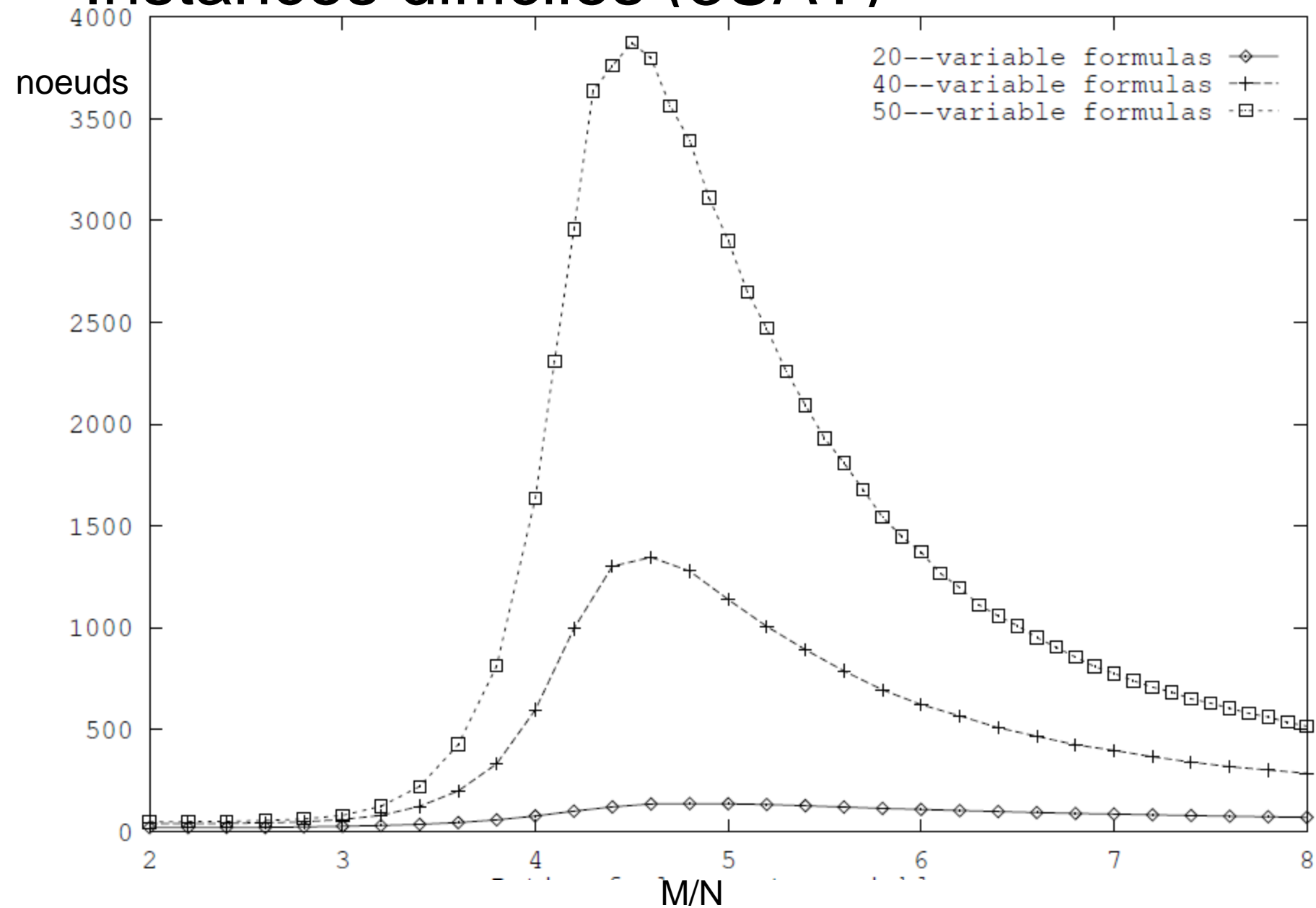
N = nombre de symboles

- Les instances difficiles figurent dans le voisinage de $M/N \approx 4.3$ (point critique)

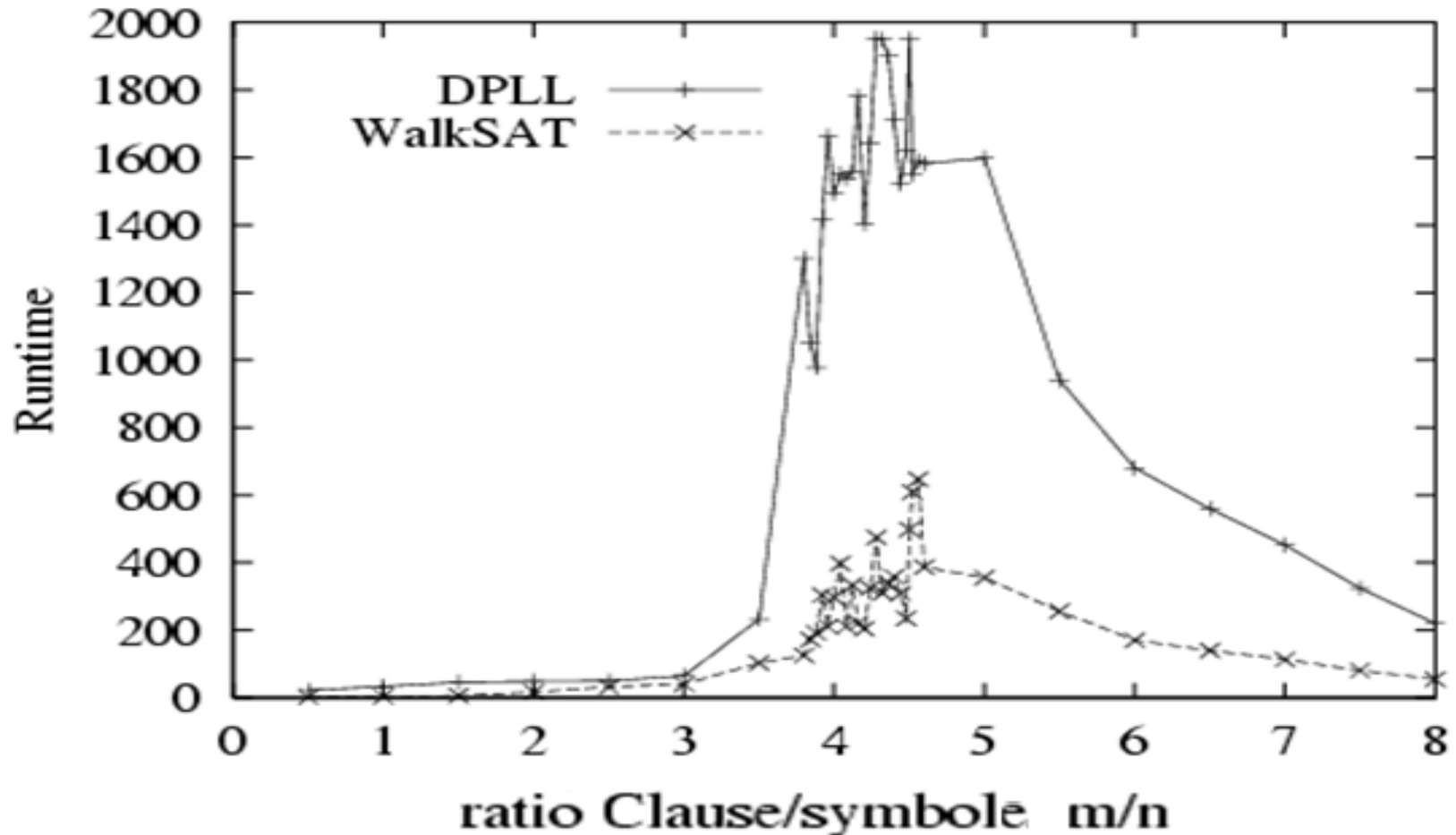
Instances difficiles



Instances difficiles (3SAT)



Instances difficiles (3SAT)



Temps median pour 100, instances 3-FNC aléatoires, $n = 50$

Instances industrielles

The instance `bmc-ibm-6.cnf`, IBM LSU 1997:

p cnf 51639 368352

—1 7 0

—1 6 0

—1 5 0

—1 —4 0

—1 3 0

—1 2 0

—1 —8 0

—9 15 0

—9 14 0

—9 13 0

—9 —12 0

—9 11 0

—9 10 0

—9 —16 0

—17 23 0

—17 22 0

i.e. $((\neg x_1) \text{ or } x_7)$
and $((\neg x_1) \text{ or } x_6)$
and ... etc.

- ✓ 15000 pages de clauses
- ✓ 50000 variables booléennes

Le solveur MiniSat (Een&Sorensson) résout l'instance en 2 seconds.

Instances industrielles

exemple : post-cbmc-zfcp-2.8-u2.cnf

p cnf 11 483 525 (vars) 32 697 150 (clauses)

1 -3 0

2 -3 0 $\leftarrow x_1 = \wedge(x_2, x_3)$

-1 -2 3 0

...

...

-11482897 -11483041 -11483523 0

11482897 11483041 -11483523 0

11482897 -11483041 11483523 0

$\leftarrow (x_3 \Leftrightarrow x_2 \Leftrightarrow x_3)$

-11482897 11483041 11483523 0

-11483518 -11483524 0

-11483519 -11483524 0

-11483520 -11483524 0

-11483521 -11483524 0

$\leftarrow x_6 = \wedge(x_7, x_8, x_9, x_{10}, x_{11}, x_{12})$

-11483522 -11483524 0

-11483523 -11483524 0

11483518 11483519 11483520 11483521 11483522 11483523 11483524 0

-8590303 -11483524 -11483525 0

8590303 11483524 -11483525 0

8590303 -11483524 11483525 0

$\leftarrow (x_{13} \Leftrightarrow x_{14} \Leftrightarrow x_{15})$

-8590303 11483524 11483525 0

-11483525 0

Résolu en moins d'une minute [Carla Gomes]

Solveurs Sat modernes

- 1. Phénomène de queues lourdes (Heavy tailed phenomena): dû aux mauvais choix associés aux variables booléennes [Gomes et al, 97]
 - Solution : redémarrage aléatoire
- 2. apprentissage des clauses de conflits: l'ajout de certaines clauses peut stopper la recherche dans des zones inutiles [Marques Silva et al. 96]
- 3. tri de variable selon leur activité: [Brisoux et al, 99], [Moskewicz et al, 01] :Heuristiques efficaces pour la simplification du problème
- 4. littéraux observés (Watched literals): [H. Zhang et al, 97], [Moskewicz et al, 01] accélérer la propagation unitaire
- 5. Symétries
- 6. parallélisme

Choix des variables de divisions

- Heuristique MOMS (Maximal Occurrence in clauses of Minimal Size) [JT96]
- Variable State Independent Decaying Sum (VSIDS)
 - [MMZ+01] À chaque variable on associe un score qui est incrémenté chaque fois que cette variable fait partie d'une clause de conflit
 - On encourage le choix de variables impliqués dans les conflits récents

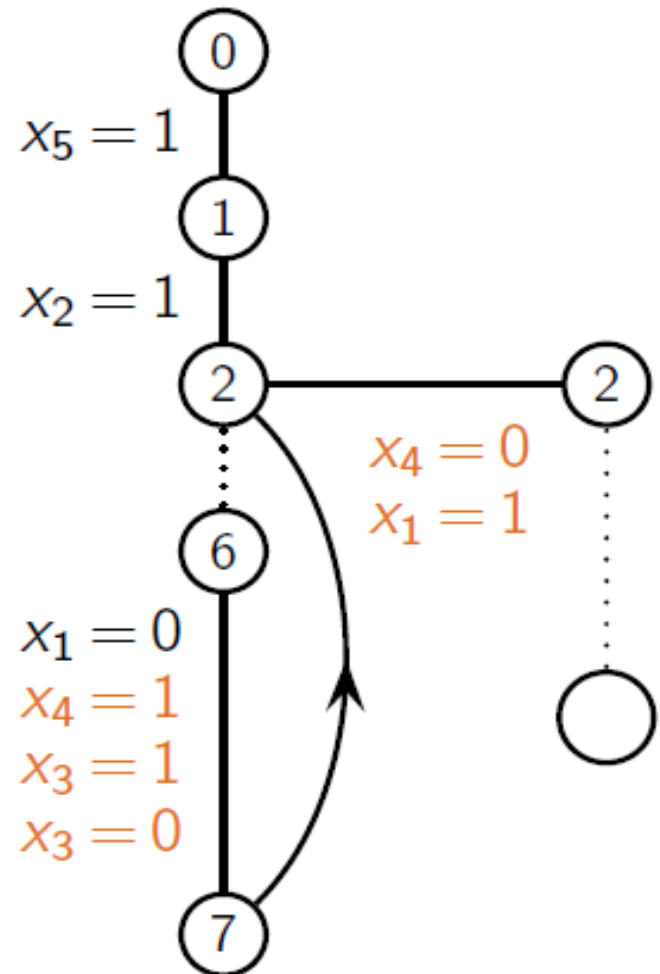
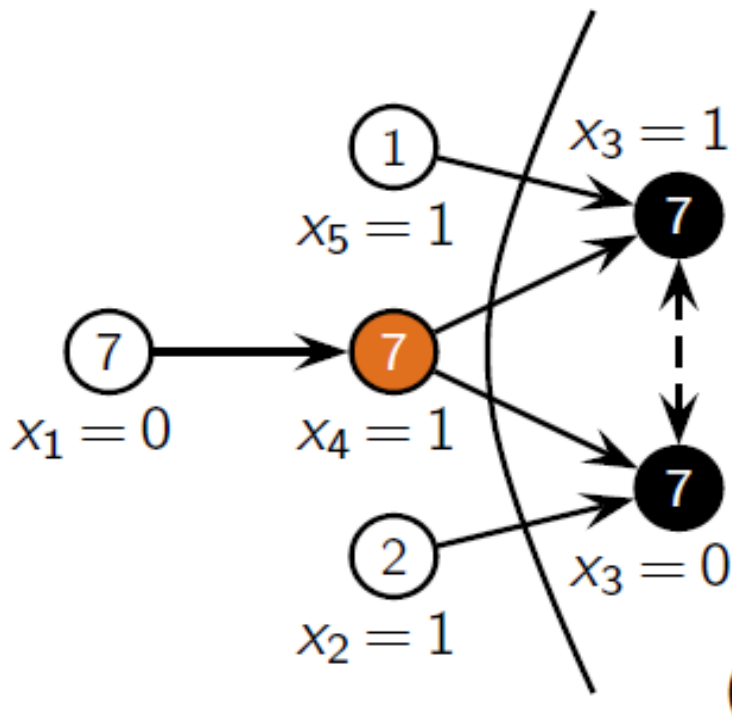


Apprentissage de clauses

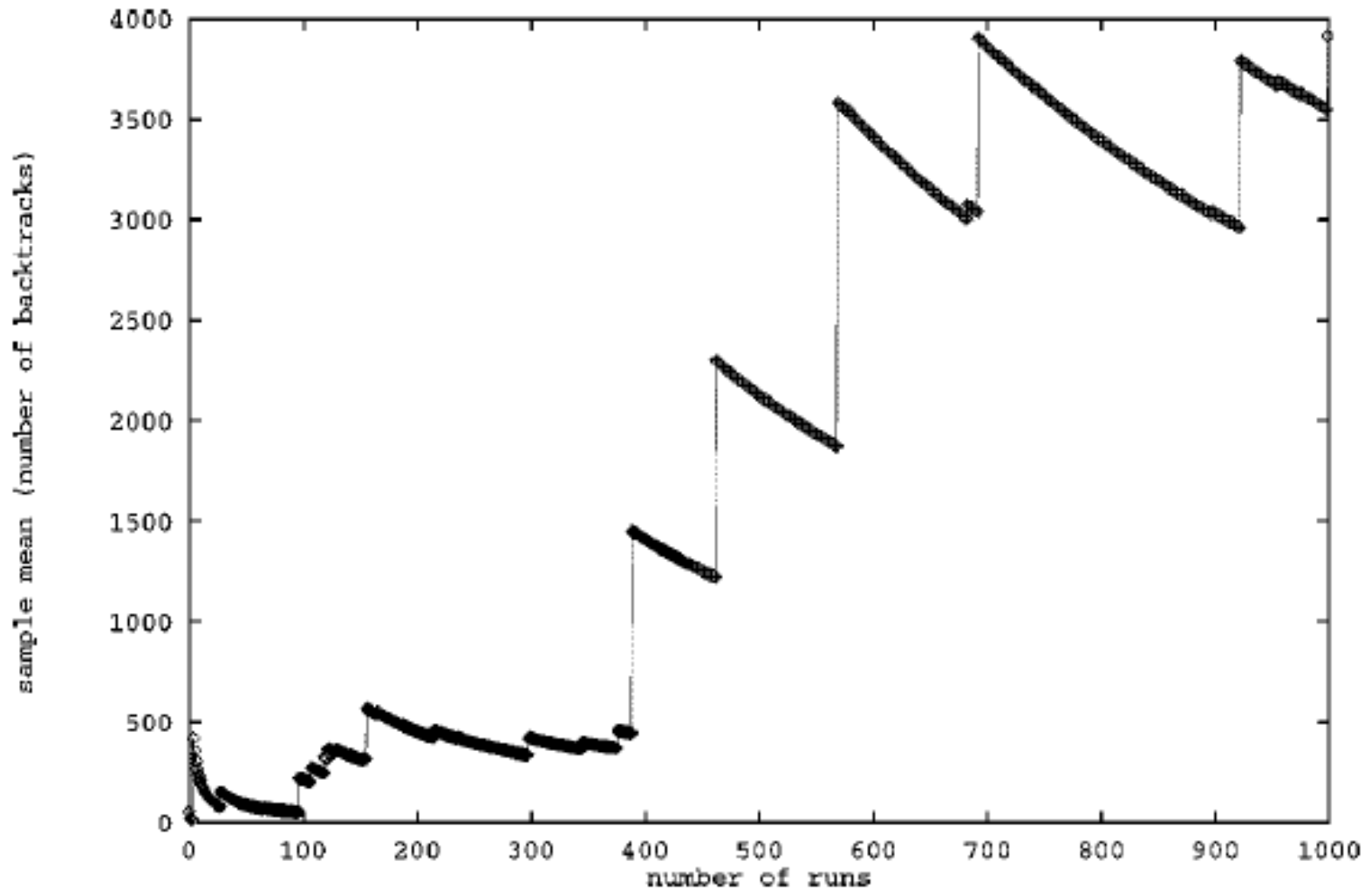
- But: élaguer l'arbre de recherche le plutôt possible
- Ajouter une nouvelle clause pour stopper les recherches inutiles dans le futur
- Faire des retours arrières non chronologiques (backjumping)

Apprentissage de clauses

$$\begin{aligned}
 &(\bar{x}_1 \vee x_4) \wedge \\
 &(x_3 \vee \bar{x}_4 \vee \bar{x}_5) \wedge \\
 &(\bar{x}_3 \vee \bar{x}_2 \vee \bar{x}_4) \wedge \\
 &\mathcal{F}_{\text{extra}}
 \end{aligned}$$



Phénomène des queues lourdes (heavy tailed phenomena)





Redémarrage aléatoire

- Eviter d'être stagné dans une branche (sous-arbre) qui n'est pas satisfaisable
- Avec un redémarrage aléatoire, il est possible qu'on commence à assigner les variables les plus critiques (backdoors)
- par conséquent, ce qui reste est facile à résoudre → le temps d'exécution est assez court
- le problème de longues queues est éliminé

Seuil de redémarrage (cut-off)

- On compte le nombre de backtracking (ou nœuds) depuis le dernier redémarrage
- Politique géométrique
- $X_i = 1.5 \times X_{i-1}$
- Politique Arithmétique:
- $X_i = X_{i-1} + 16000$
- Serie Luby: e.g. 100, 100, 200, 100, 100, 200, 400,

symétries

- Il existe des variables qui prennent la même valeur dans n'importe quel model
- On les appellent variables symétriques
- Une symétrie est une permutation de littéraux qui laisse invariant la formule.

symétries

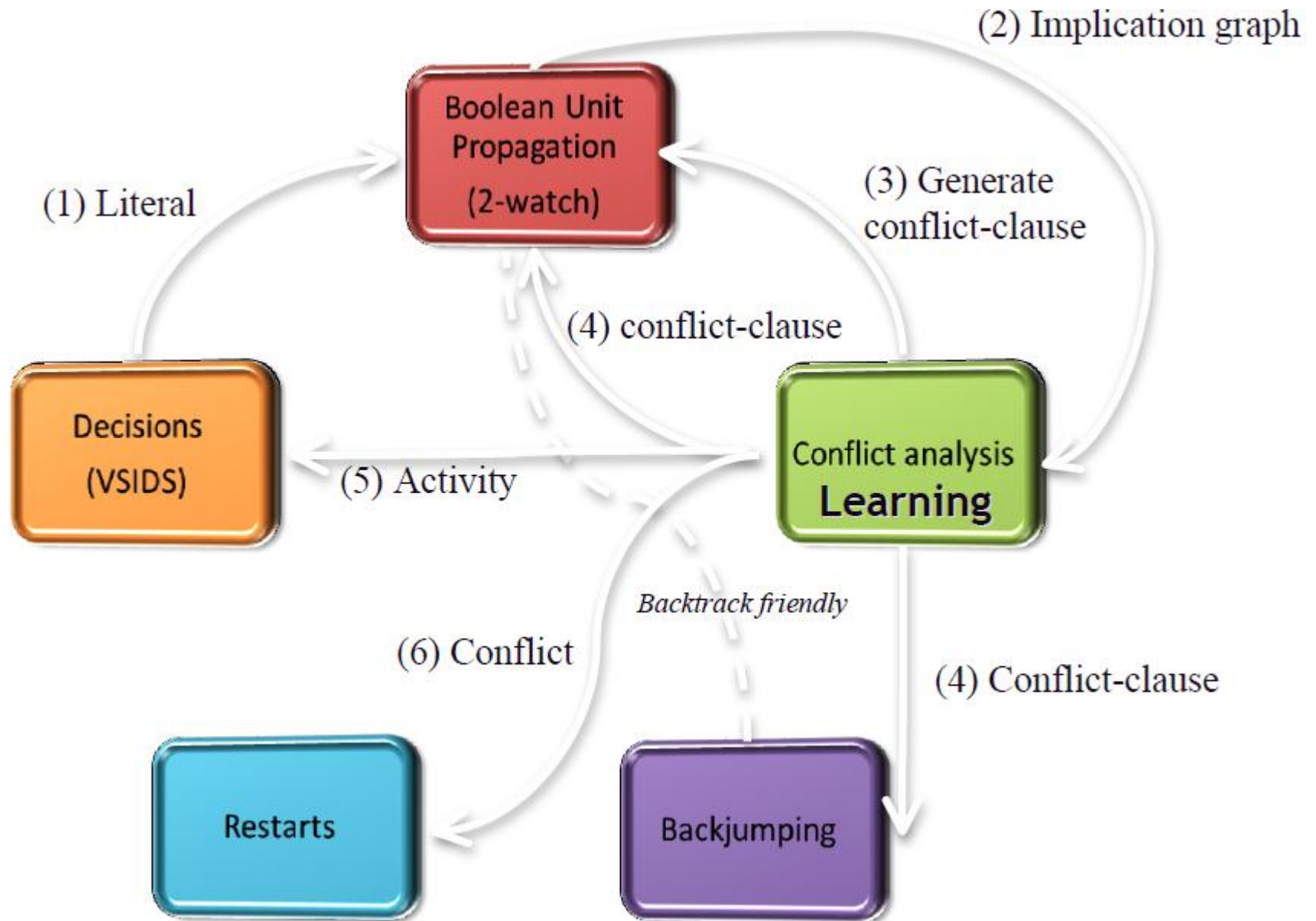
- $F = a \vee b \vee c \wedge \neg a \vee b \wedge \neg b \vee c \wedge \neg c \vee a \wedge \neg a \vee \neg b \vee \neg c$
- $\sigma = \{(a, \neg a)(b, \neg c)(c, \neg b)\}$
- Rechercher les variables symétriques pour chaque nœud apporte un gain considérable (élagage de l'espace de recherche).



parallélisme

- Assigner les variables et affecter les nœuds créés aux processeurs.
- Exploitation des multi-coeurs

Solveurs Sat modernes





References utiles

- ✓ SAT competition:
<http://www.satcompetition.org/>
- ✓ I International Conference on Theory and Application of Satisfiability Testing (SAT)

Propriétés fondamentales

■ Correction

- Toute formule prouvable est valide

- si $\vdash A$ alors $\models A$

■ Complétude

- Toute formule valide est prouvable

- si $\models A$ alors $\vdash A$

■ Décidabilité

- Pour toute formule $A \in \text{Prop}$, il y a un algorithme qui vérifie en un temps fini si A est un théorème (formule valide) ou non.



FIN