Programmation Par Contraintes

Complexité des problèmes de contraintes

Florent Madelaine Éric Nespoulos et David Savourey

MPRO

Théorie et Pratique de SAT

Année 2015 - 2016

Administration

Projet et groupes

- Se fait forcément en binôme.
- Vous êtes 22 inscrits. A priori pas besoin de faire des groupes de taille impaire.
- Si vous pensez que c'est nécessaire : envoyez svp un mail à David avec la composition de ce groupe spécial de taille impaire (monôme ou trinôme).
- Ce groupe spécial est forcément unique.

Projet et le rendu

- on attend un rapport synthétique (3/4 pages maxi)

Plan des cours

Partie 1: Théorie et pratique

Partie 2: Les cas polynomiaux du théorème de Schaefer

Partie 3: Approche algébrique et théorème de Schaefer

Partie 4: Les cas NP-complets du théorème de Schaefer

Partie 5: Solveurs modernes

Première partie I

Théorie et pratique

Un peu d'histoire

réconcilier théorie et pratique?

classification de la complexité (conjecture de la dichotomie)

Dichotomie

Restrictions de Sat

Théorème de Schaefer

un peu d'histoire

Pourquoi la PPC?

8			4		6			7
						4		
	1					6	5	
5		9		3		7	8	
				7				
	4	8		2		1		3
	5	2					9	
		1						
3			9		2			5

- Problème générique : modélisation ≪ facile ≫
- Idée résolution : search + filtrage.
- En pratique pas facile à utiliser!

Un cas particulier + ancien : SAT

- Idée de résolution similaire : DPLL
- 1962 Martin Davis, Hilary Putnam, George Logemann and Donald W. Loveland)
- (différent idée initiale des 2 premiers basée sur la résolution)

PPC vs SAT

Sat solvers

- Solveurs SAT plus anciens que solveurs CSPs
- Gros progrès depuis 15 à 20 ans
- Exemple idée importante : apprentissage clause + backjumping
  - CDCL solvers (conflict driven clause learning)
  - Routine : 10^6 variables en quelques secondes.
- Application centrale : *approximate verif of timed system* $\rightsquigarrow$ *Bounded Model checking*

Biblio

Sharad Malik, Lintao Zhang: *Boolean satisfiability from theoretical hardness to practical* success. Commun. ACM 52(8): 76-82 (2009)

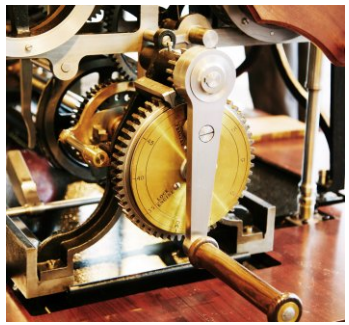
PPC vs Sat II

PPC

- Solveurs CSPs ”plus proches” des problèmes
- Apparente simplicité à modéliser vs Technicité à fabriquer un modèle booléen efficace pour un problème.
- Progrès pas en isolation : l’idée de CDCL est en fait une vieille idée de PPC.

Rêve de la machine miraculeuse!

- λ modélise et résoud son problème.
- Réalité : travail d’expert connaissant le solveur.



Et en pratique?

- Plutôt bons sur les aspects discrets
- Moins pour d'autres aspects.
- En pratique interaction avec des solveurs dédiés.

Exemples pour SAT

- package dependency management for the OpenSuSE Linux distribution
- autonomous controller for NASA's Deep Space One spacecraft

Exemples de telles interactions

- SAT modulo theory (SMT solvers) très utilisés en vérification.
- En RO modéliser une partie en PLNE, une partie en PPC (le second sert à générer des colonnes pour les premier) (cf exemple Éric)

Et la théorie dans tout ça?

Complexité

- Théorème de Cook - Levin :
  Sat est NP-complet.
- Conjecture : P $\neq$ NP
- En pratique : pas d'algo polynomial connu.
- Autre conjecture :
  ETH pour SAT by Impagliazzo, Paturi & Zane (2001)

sad line of Programmers (Garey & Johnson's book on NP-completeness)

etc

- Millenium prize.
- trivia : preuves fausses de P=NP (ou le contraire) régulières.

Un peu d'histoire

réconcilier théorie et pratique?

classification de la complexité (conjecture de la dichotomie)

Dichotomie

Restrictions de Sat

Théorème de Schaefer

Incohérence

Théorie vs Pratique

- Complexité dit : dur
- Mais Sat solveurs marchent!

Approche très différente

- Complexité : (pire des cas) séquence d'entrées, asymptotiques.
- Pratique sur les solveurs : 1 problème à modéliser et résoudre.

Question

Pas la bonne théorie?

Biblio

Moshe Y. Vardi: *On P, NP, and computational complexity*. Commun. ACM 53(11): 5 (2010)

Réconcilier théorie et pratique

Question

Quelle est la bonne théorie?

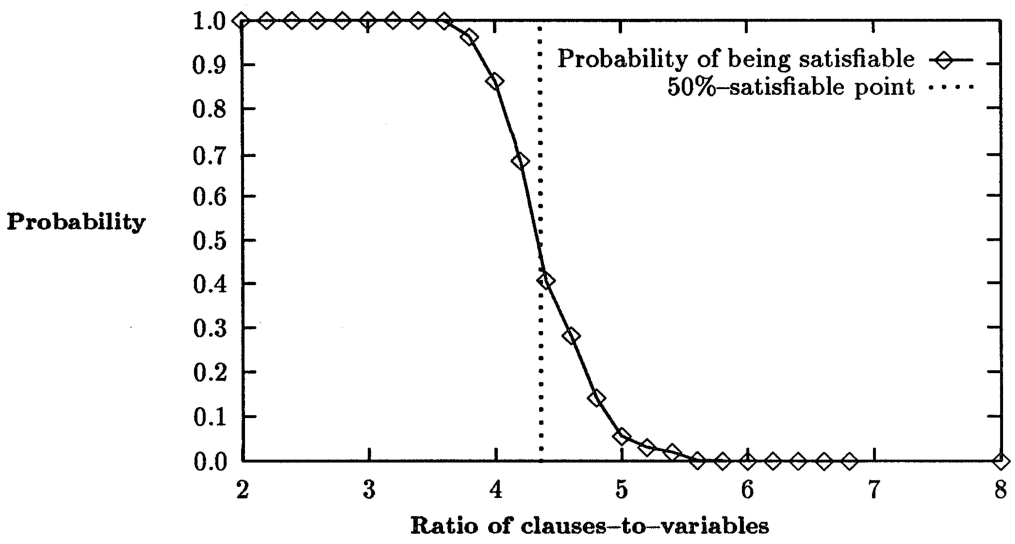
Complexité en moyenne?

- Pas clair.
- Que prend on comme distribution des instances?
- Distribution uniforme pas du tout semblable à instances qu'on veut résoudre en pratique.

Transition de phases

Une histoire de densité?

- Phénomènes de seuil
- Instances de k-sat avec n variables et m clauses.
- pour $k = 3$, quand $r = \frac{m}{n}$ est autour de 4,27 (expérimental)



biblio

- Peter Cheeseman, Bob Kanefsky, William M. Taylor: *Where the Really Hard Problems Are*. IJCAI 1991: 331-340
- David G. Mitchell, Bart Selman, Hector J. Levesque: *Hard and Easy Distributions of SAT Problems.* AAAI 1992: 459-465

transitions vs complexité?

Pas clair

- pour k-Sat, la transition sat/transition/UnSat devient facile / vraiment difficile / dur mais pas trop
- les résultats expérimentaux ne semblent pas en accord avec la théorie (mais la théorie travaille à la limite!)
- pour des restrictions de Sat, il n'y a pas forcément des liens clairs entre difficulté et transitions.

D'autres résultats

- compétition pour démontrer la valeur du seuil.
- plusieurs phases pour la répartition des solutions (topologie = distance de Hamming).

Ce qu'on fait

Approche pratique.

- Benchmarks
- Compétitions

links

- `http://www.satlive.org`
- `http://www.csplib.org/`
- `https://www.minizinc.org/challenge.html`

Ce qu'on fait

Approche théorique

Restriction du problème générale ayant des algos efficaces

- graphe des contraintes arborescents (liens requêtes BdD)
- langages restreints
- hybrides

Liens avec pratique?

- complexité paramétrée / (backdoor)
- souvent : petit nombre de variables responsables de la difficulté sur 1 instance "industrielle".
- Solveurs pratiques arrivent à trouver ses backdoor via l'apprentissage et le backjumping.

Suite du cours

se concentrer sur Sat

- résultats de complexité (joli théorie même si pas forcément la bonne théorie)
- introduction au fonctionnement des solvers modernes (CDCL).

Un peu d'histoire

réconcilier théorie et pratique?

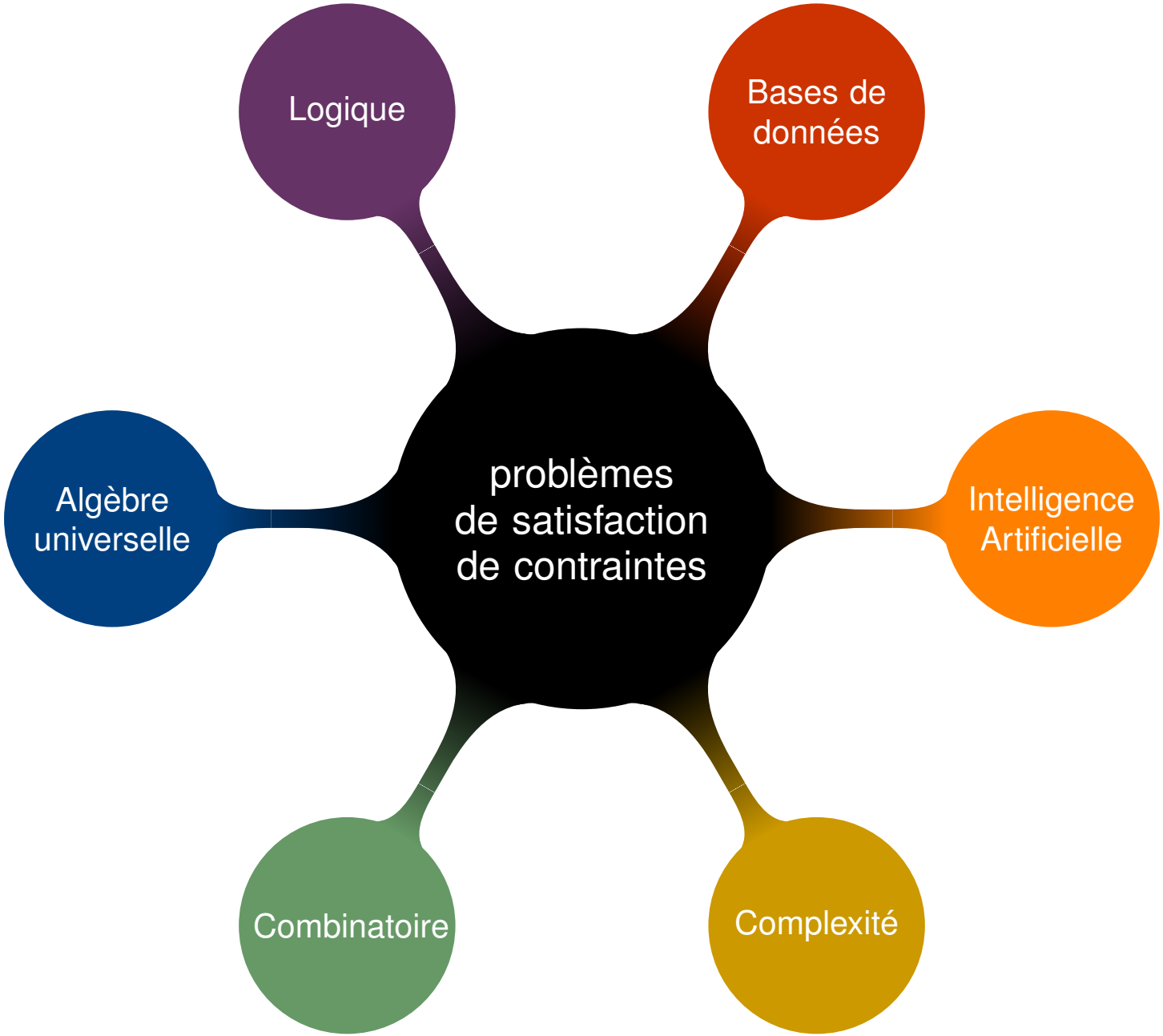
classification de la complexité (conjecture de la dichotomie)

Dichotomie

Restrictions de Sat

Théorème de Schaefer

ubiquité des CSP



Une définition combinatoire des problèmes de contraintes

On peut formaliser le problème de contraintes général comme un problème d'homomorphisme entre deux structures relationnelles A et B et les restrictions correspondent par exemple à fixer la seconde structure B qui code le langage des contraintes.

Dans le cas de graphes non orientés,

- lorsque le langage des contraintes contient seulement la relation $\neq$ (une clique), on retrouve le problème de k-colorabilité d'un graphe; et
- si on prend un graphe quelconque $B := H$, on a le problème de H-coloring.

Le théorème de Hell et Nešetřil établit que le H-coloring suit une dichotomie.

Complexité des problèmes de contraintes et algèbre universelle

La complexité des problèmes de contraintes non-uniformes [1], semble s'expliquer par des propriétés algèbriques des langages de contraintes.
On verra comment on peut s'appuyer sur le treillis des clones booléens pour démontrer le théorème de Shaeffer, qui établit une dichotomie dans le cas Booléen.
On peut aussi montrer que le théorème de Hell et Nešetřil s'explique algébriquement.

Intuition:

- expressivité d'un langage de contrainte $\iff$ fonctions préservant les relations de ce langage
- Préservation par certaines fonctions particulières $\implies$ existence algorithme polynomial.

1. Les restrictions du problème général où l'on fixe un langage de contrainte

Complexité des problèmes de contraintes et combinatoire

La complexité des problèmes de contraintes uniformes [2], s'explique précisément par l'existence de décomposition arborescente de largeur bornée du graphe des contraintes.

Lorsque de telles décomposition existent, on peut résoudre le problème de décision en utilisant le filtrage (k-cohérence). Plus généralement, on utilise la programmation dynamique.

2. Les restrictions du problème général où l'on restreint le graphe des contraintes, i.e. la structure des contraintes sur les variables, quelle que soit le langage des contraintes

Définition IA « classique »

Une instance du problème de satisfaction de contraintes CSP est un triplet $(\text{Var}, \text{Dom}, \mathcal{C})$ où

- Var est un ensemble de variables,
- Dom est un ensemble de valeurs et
- $\mathcal{C}$ est un ensemble de contraintes:
  - chaque contrainte étant de la forme $(v_{i_1}, \ldots, v_{i_r}, R)$ où r est l'arité de la contrainte et $R \subseteq \text{Dom}^r$.

Une solution est une application des variables aux valeurs qui satisfait simultanément toutes les contraintes.

Un cas particulier important

Le cas booléen

SAT correspond à CSP avec un domaine booléen. Comme l'entrée est codée de manière différente (pas sous forme clausale), on parle de *Generalized Satisfiability*

Exemple

clause

$x \vee y \vee \bar{z}$

contrainte

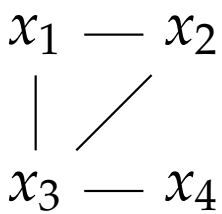
- (x, y, z)
- $\{0, 1\}^3 \setminus \{(0, 0, 1)\}$

Modéliser 3-col

Pour un graphe $G := (V, E)$ on a l'instance de CSP suivante

- variables Var $:= V$,
- valeurs Dom $:= \{1, 2, 3\}$ et
- contraintes $\mathcal{C} := \{(x_i, x_j, \neq) : \text{ pour chaque } (x_i, x_j) \in E\}$.

Exemple



- instance: Var $= \{x_1, x_2, x_3, x_4\}$, Dom $= \{1, 2, 3\}$ et $\mathcal{C} = \{((x_1, x_2), \neq), ((x_1, x_3), \neq), ((x_3, x_2), \neq), ((x_3, x_4), \neq)\}$
- une solution: $x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 1$

Deux structures sous-jacentes

Notre exemple

- instance: Var $= \{x_1, x_2, x_3, x_4\}$, Dom $= \{1, 2, 3\}$ et $\mathcal{C} = \{((x_1, x_2), \neq), ((x_1, x_3), \neq), ((x_3, x_2), \neq), ((x_3, x_4), \neq)\}$
- une solution: $x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 1$

Deux structures

Variables $\mathcal{A}$ Valeurs $\mathcal{B}$

homomorphisme : $x_1 \mapsto 1, x_2 \mapsto 2, x_3 \mapsto 3, x_4 \mapsto 1$

Homomorphisme

Vous avez dû rencontrer la notion d'homomorphisme en algèbre pour des groupes, des anneaux. De manière générale, il s'agit d'une application h (du domaine) d'une structure A vers (le domaine d')une structure B qui préserve les propriétés algébriques de ces structures.
Dans le cas de structures relationnelles, on a:

- pour tout symbole R, pour tout tuple $a_1, a_2, \ldots, a_r$ d'éléments de A, si $(a_1, a_2, \ldots, a_r) \in R^A$ alors $(h(a_1), h(a_2), \ldots, h(a_r)) \in R^B$

En particulier, pour le cas des graphes non orientés (une seule relation binaire qui est symmétrique) un homomorphisme envoie une arête du graphe A sur une arête du graphe B. Si les graphes sont orientés (une seule relation binaire) alors on envoie un arc sur un arc en préservant l'orientation.

CSP vu comme un problème d'homomorphisme

- instance : une paire de structures relationnelles similaires $(\mathcal{A}, \mathcal{B})$
- question : existe-t-il un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$?

où $\mathcal{A}$ représente la structure des contraintes et
$\mathcal{B}$ représente le language des contraintes

Exemple (3-col comme un homomorphisme dans K_3)

homomorphisme : $x_1 \mapsto 1, x_2 \mapsto 2, x_3 \mapsto 3, x_4 \mapsto 1$

Un peu d'histoire

réconcilier théorie et pratique?

classification de la complexité (conjecture de la dichotomie)

Dichotomie

Restrictions de Sat

Théorème de Schaefer

Complexité des problèmes de contraintes

- classe des problèmes NP-complet: les plus difficiles de la classe NP

  Exemples: SAT, 3-colorabilité
- Pour NP-complet, lire “non polynomial” (conjecture $P \neq NP$).
- En général décider si des contraintes peuvent être satisfaites simultanément est NP-complet.

Complexité des problèmes de contraintes

- classe des problèmes NP-complet: les plus difficiles de la classe NP

  Exemples: SAT, 3-colorabilité
- Pour NP-complet, lire “non polynomial” (conjecture $P \neq NP$).
- En général décider si des contraintes peuvent être satisfaites simultanément est NP-complet.

Complexité des problèmes de contraintes

- classe des problèmes NP-complet: les plus difficiles de la classe NP

  Exemples: SAT, 3-colorabilité
- Pour NP-complet, lire “non polynomial” (conjecture $P \neq NP$).
- En général décider si des contraintes peuvent être satisfaites simultanément est NP-complet.

Problématique

Nous voulons savoir quelles restrictions du langage des contraintes ou du graphe des contraintes garantissent une réponse en un temps raisonnable

Complexité des problèmes de contraintes

- classe des problèmes NP-complet: les plus difficiles de la classe NP

  Exemples: SAT, 3-colorabilité
- Pour NP-complet, lire “non polynomial” (conjecture $P \neq NP$).
- En général décider si des contraintes peuvent être satisfaites simultanément est NP-complet.

Problématique

Nous voulons savoir quelles restrictions du langage des contraintes ou du graphe des contraintes sont dans P

Intérêt de la définition en tant que problème d'homomorphisme

Définition de CSP en terme d'homomorphisme

- instance : une paire de structures relationnelles similaires $(\mathcal{A}, \mathcal{B})$
- question : existe-t-il un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$?

Graphe des contraintes

(graphe associé à la) structure $\mathcal{A}$.

Langage des contraintes

Relations de la structure $\mathcal{B}$

Deux façons naturelles de restreindre le problème.

Intérêt de la définition en tant que problème d'homomorphisme

Définition de CSP en terme d'homomorphisme

- instance : une paire de structures relationnelles similaires $(\mathcal{A}, \mathcal{B})$
- question : existe-t-il un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$?

Graphe des contraintes

(graphe associé à la) structure $\mathcal{A}$.

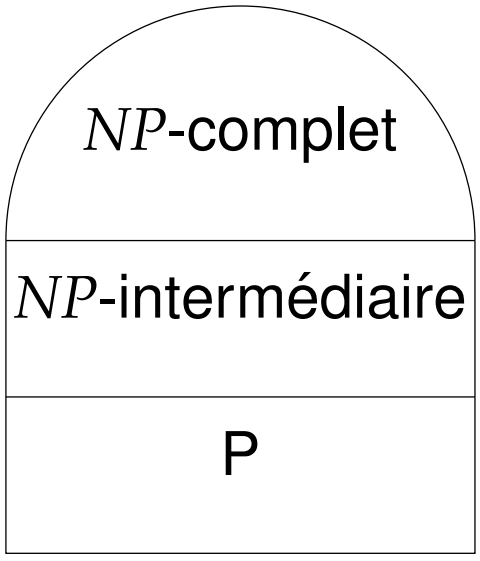
Langage des contraintes

Relations de la structure $\mathcal{B}$

Deux façons naturelles de restreindre le problème.

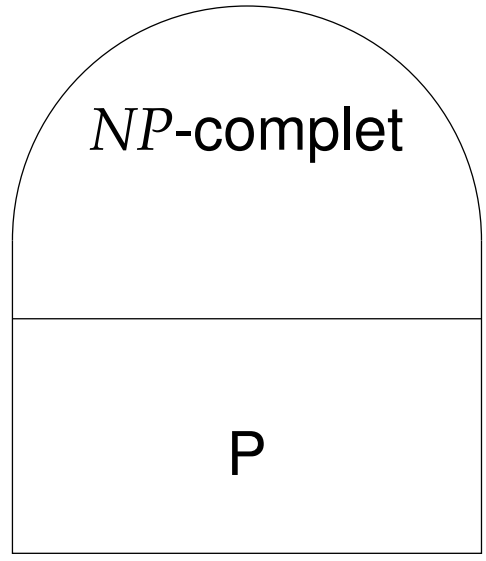
Conjecture de la dichotomie

Monde réel [3]



Théorème de Ladner [’75]

Monde CSP



Conjecture Feder et Vardi [’93]

Résultats de dichotomie pour SAT [Schaefer ’78] H-coloriage [Hell et Nešetřil ’90] ... pour 3 valeurs [Bulatov ’02] ...

3. *provisio* $P \neq NP$

Cadre de notre travail de classification

- 1 instance $\neq$ 1 problème
- On s'intéresse au problème de décision (réponse oui/non, ici sat/unsat).
- Problème de décision = ensemble d'instances « oui » (satisfaisables)
- Modèle de complexité : pire des cas (étude asymptotique)
- Notez : nombre fini d'instances pas intéressant ($\rightsquigarrow O(1)$).

Cadre de notre travail de classification

- 1 instance $\neq$ 1 problème
- On s'intéresse au problème de décision (réponse oui/non, ici sat/unsat).
- Problème de décision = ensemble d'instances « oui » (satisfaisables)
- Modèle de complexité : pire des cas (étude asymptotique)
- Notez : nombre fini d'instances pas intéressant ($\rightsquigarrow O(1)$).

Cadre de notre travail de classification

- 1 instance $\neq$ 1 problème
- On s'intéresse au problème de décision (réponse oui/non, ici sat/unsat).
- Problème de décision = ensemble d'instances « oui » (satisfaisables)
- Modèle de complexité : pire des cas (étude asymptotique)
- Notez : nombre fini d'instances pas intéressant ($\rightsquigarrow O(1)$).

Cadre de notre travail de classification

- 1 instance $\neq$ 1 problème
- On s'intéresse au problème de décision (réponse oui/non, ici sat/unsat).
- Problème de décision = ensemble d'instances « oui » (satisfaisables)
- Modèle de complexité : pire des cas (étude asymptotique)
- Notez : nombre fini d'instances pas intéressant ($\rightsquigarrow O(1)$).

Cadre de notre travail de classification

- 1 instance $\neq$ 1 problème
- On s'intéresse au problème de décision (réponse oui/non, ici sat/unsat).
- Problème de décision = ensemble d'instances $\ll$ oui $\gg$ (satisfaisables)
- Modèle de complexité : pire des cas (étude asymptotique)
- Notez : nombre fini d'instances pas intéressant ($\rightsquigarrow O(1)$).

CSP uniformes et non-uniformes

Discussion + tableau.

Le théorème de Schaefer concerne des restrictions du langage de contrainte.

CSP uniformes et non-uniformes

Discussion + tableau.

Le théorème de Schaefer concerne des restrictions du langage de contrainte.

Sat, Generalized Sat et restrictions

2 façons de définir SAT.

- classique : version propositionnelle (CNF) il s'agit du problème du théorème de Cook.
- comme un CSP Booléen (generalized Sat), des contraintes (relations) remplacent les clauses.

Restrictions de ces 2 cas :

- nombre de variables par clause / arité des contraintes
- plus généralement, certains type de clauses / certaines relations.

Exemples

- k-Sat (arité k)
- Horn-Sat c'est-à-dire au plus un litéral positif par clause (version généralisée : plus tard).

Sat, Generalized Sat et restrictions

2 façons de définir SAT.

- classique : version propositionnelle (CNF) il s'agit du problème du théorème de Cook.
- comme un CSP Booléen (generalized Sat), des contraintes (relations) remplacent les clauses.

Restrictions de ces 2 cas :

- nombre de variables par clause / arité des contraintes
- plus généralement, certains type de clauses / certaines relations.

Exemples

- k-Sat (arité k)
- Horn-Sat c'est-à-dire au plus un litéral positif par clause (version généralisée : plus tard).

Sat, Generalized Sat et restrictions

2 façons de définir SAT.

- classique : version propositionnelle (CNF) il s'agit du problème du théorème de Cook.
- comme un CSP Booléen (generalized Sat), des contraintes (relations) remplacent les clauses.

Restrictions de ces 2 cas :

- nombre de variables par clause / arité des contraintes
- plus généralement, certains type de clauses / certaines relations.

Exemples

- k-Sat (arité k)
- Horn-Sat c'est-à-dire au plus un litéral positif par clause (version généralisée : plus tard).

Exo : complexité de k-SAT

1. Montrez que 1-SAT peut être résolu en temps polynomial.
2. Montrez que 2-SAT peut être résolu en temps polynomial.
3. Montrez que si 3-SAT peut être résolu en temps polynomial alors SAT peut l'être aussi.

GENERALIZED SATISFIABILITY

Nous allons étudier le cas des CSP non uniformes dont le langage des contraintes est un domaine à 2 valeurs. On parle de GENERALIZED SATISFIABILITY. On ne voit plus les instances sous forme clausale comme pour SAT.

- paramètre : une structure relationelle $\mathcal{B}$ de domaine $\{0, 1\}$.
- instance : une structure relationnelle similaires $\mathcal{A}$
- question : existe-t-il un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$?

Un nouveau problème

Puisque on verra que ce qui importe ce sont les relations Γ de la structure $\mathcal{B}$, on paramétrise le problème par ces dernières plutôt que la structure $\mathcal{B}$.

GENERALIZED-SAT (CSP($\mathcal{B}$))

- paramètre : une structure relationelle $\mathcal{B}$ de domaine $\{0, 1\}$.
- instance : une structure relationnelle similaires $\mathcal{A}$
- question : existe-t-il un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$?

GENERALIZED-SAT (SAT(Γ))

- paramètre : un ensemble fini de relations Γ.
- instance : une conjonction φ d'atomes positifs $R(\bar{x})$, où R est un symbole d'une relation de Γ.
- question : est-ce-que φ est satisfaisable?

Un nouveau problème

Puisque ce qui importe ce sont les relations Γ de la structure $\mathcal{B}$, on paramétrise le problème par ces dernières plutôt que la structure $\mathcal{B}$.

GENERALIZED-SAT (SAT(Γ))

- paramètre : un ensemble fini de relations Γ.
- instance : une conjonction φ d'atomes positifs $R(\bar{x})$, où R est un symbole d'une relation de Γ.
- question : est-ce-que φ est satisfaisable?

- On verra qu'on peut même considérer des ensembles Γ qui sont arbitraires.
- Pour l'instant, on considère que Γ est fini.

Un nouveau problème

Puisque ce qui importe ce sont les relations Γ de la structure $\mathcal{B}$, on paramétrise le problème par ces dernières plutôt que la structure $\mathcal{B}$.

GENERALIZED-SAT (SAT(Γ))

- paramètre : un ensemble (possiblement non fini) de relations Γ.
- instance : une conjonction φ d'atomes positifs $R(\bar{x})$, où R est un symbole d'une relation de Γ.
- question : est-ce-que φ est satisfaisable?

- On verra qu'on peut même considérer des ensembles Γ qui sont arbitraires.
- Pour l'instant, on considère que Γ est fini.

Un nouveau problème

Puisque ce qui importe ce sont les relations Γ de la structure $\mathcal{B}$, on paramétrise le problème par ces dernières plutôt que la structure $\mathcal{B}$.

GENERALIZED-SAT (SAT(Γ))

- paramètre : un ensemble fini de relations Γ.
- instance : une conjonction φ d'atomes positifs $R(\bar{x})$, où R est un symbole d'une relation de Γ.
- question : est-ce-que φ est satisfaisable?

- On verra qu'on peut même considérer des ensembles Γ qui sont arbitraires.
- Pour l'instant, on considère que Γ est fini.

Un peu d'histoire

réconcilier théorie et pratique?

classification de la complexité (conjecture de la dichotomie)

Dichotomie

Restrictions de Sat

Théorème de Schaefer

Forme du théorème de Schaefer

- Si le langage de contrainte booléen $\mathcal{B}$ est de type
  ... 6 cas ...
  alors $\mathrm{CSP}(\mathcal{B})$ est dans P.
- Sinon $\mathrm{CSP}(\mathcal{B})$ est NP-complet.

Forme du théorème de Schaefer

- Si le langage de contrainte booléen $\mathcal{B}$ est de type Horn, dual-Horn, bijonctif, affine, 0-valide ou 1-valide alors $\text{CSP}(\mathcal{B})$ est dans P.
- Sinon $\text{CSP}(\mathcal{B})$ est NP-complet.

Exos : complexité de HORN-SAT

1. Soit φ une formule de Horn à n variables. On suppose qu'il existe une clause constituée d'un seul litéral positif x dans φ. Montrez que satisfaire φ revient à satisfaire une formule de Horn à $n-1$ variables.
2. En utilisant le fait qu'on peut écrire une clause de Horn sous forme implicative, en déduire un algorithme de résolution pour HORN-SAT.
3. Quelle est sa complexité dans le pire des cas en fonction de n (nombre de variables) et m (nombre de clauses).
4. Appliquer cet algorithme à

$$\varphi := (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_3 \vee \neg x_2 \vee \neg x_4)$$
$$\wedge (x_2 \vee \neg x_3) \wedge (\neg x_4 \vee \neg x_3) \wedge (x_3) \wedge (x_5 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3)$$

Stratégie de preuve

Cas polynomiaux

Pour les 6 cas (sauf 2 qui sont triviaux), en gros 2 méthodes:

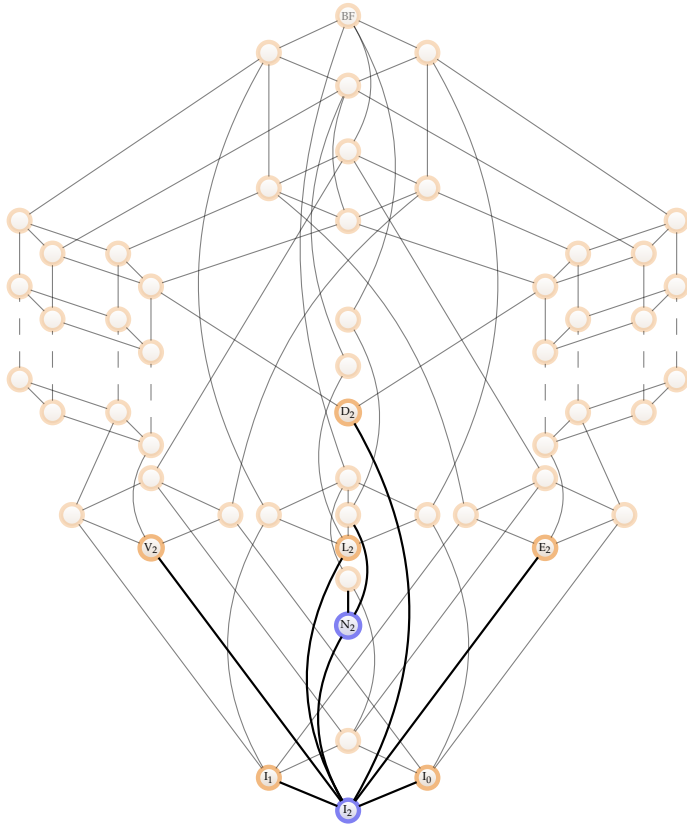
- réduction à un cas similaire de Sat (sous forme clausale) qui est dans P
- résolution directe (l'algo est souvent « le même »).

Chaque cas correspond à une propriété de clôture des relations par une fonction booléenne.

Cas difficiles

On se penche sur une notion naturelle d'expressivité d'un langage de contrainte. Ceci correspond au niveau combinatoire à la construction de « gadgets ». On peut faire une preuve directe (assez technique) que si on ne tombe pas dans les 6 cas faciles, on peut alors toujours exprimer une relation difficile (correspondant à NOT-ALL-EQUAL-3-SAT ou bien 1-IN-3-SAT). Nous allons faire une preuve indirecte en explicitant la notion d'expressivité dans le cadre de la théorie des clones.

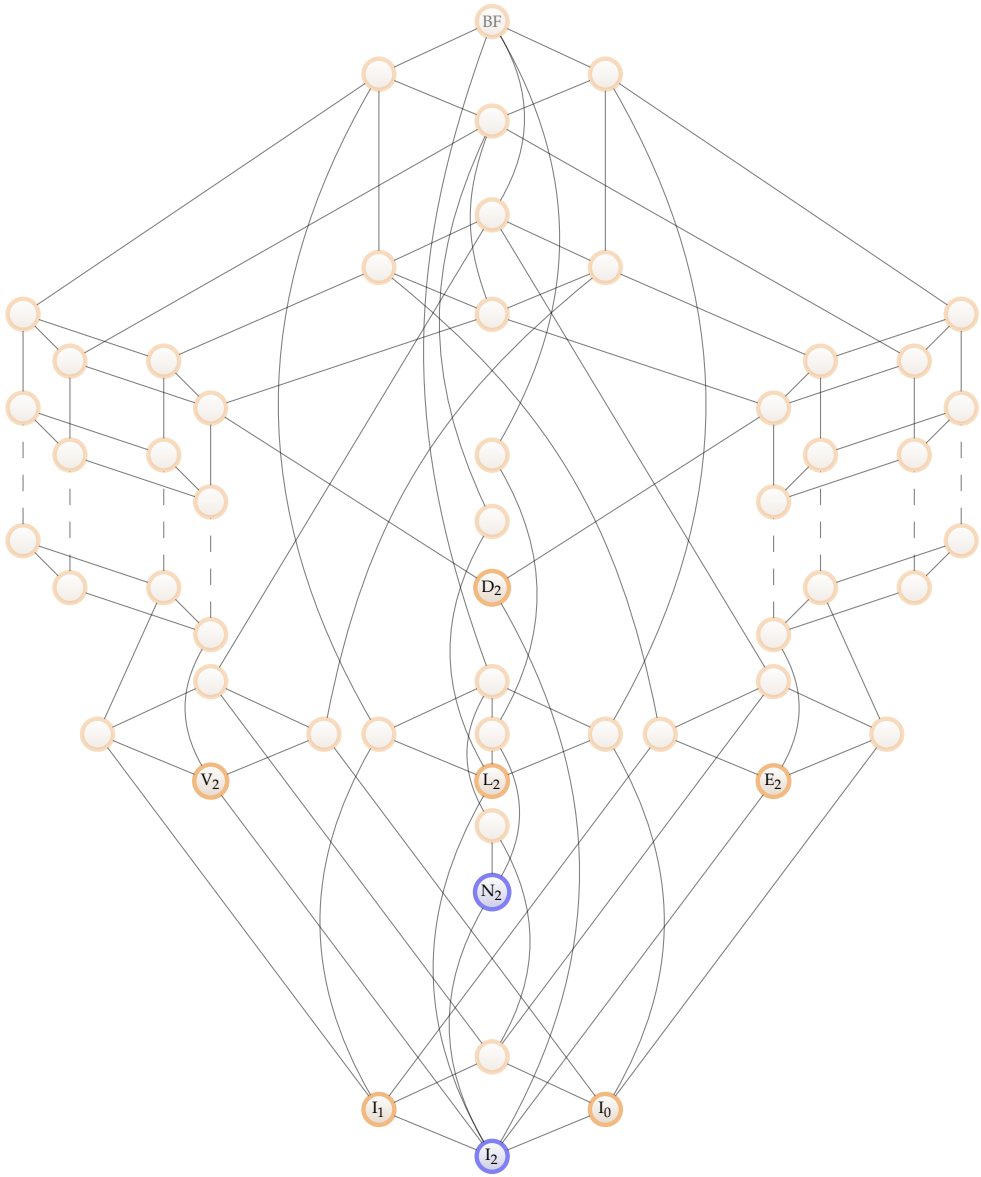
Treillis de Post



Méthodologie

- élément de ce treillis = ensemble de relations (clos pour la notion d'expressivité)
- il suffit de prouver que les ensembles en orange sont dans P et ceux en bleu sont NP-difficiles.

Treillis de Post



Exo : inter Réduction entre SAT et GENERALIZED SAT

1. Réduisez K-SAT à GENERALIZED SAT.
2. On considère le cas de GENERALIZED SAT restreint à une seule relation R d'arité r. Réduisez ce problème à SAT.

Deuxième partie II

Les cas polynomiaux du théorème de Schaefer

Plan du cours

Complexité et préservation

Nous allons voir que les langages de contraintes polynomiaux se caractérisent tous par une propriété de clôture. Les 6 cas correspondent aux 6 opérations suivantes, d'arité 1, 2 ou 3.

- c_0, c_1 (constantes)
- $\wedge, \vee$ binaires
- m, l ternaires

Définition: préservation / clôture

Une fonction booléenne f d'arité n préserve une relation R d'arité m (on dit aussi que R est close par f)

ssi

pour tous tuples $t_1 = (a_{11}, \ldots, a_{1m}) \ldots t_n = (a_{n1}, \ldots, a_{nm})$ appartenant à cette relation R, le tuple $f(t_1, \ldots, t_r)$ (on applique f composante par composante) appartient aussi à R.

$$\frac{\begin{array}{ccccc} f & & f & & f \\ (\; a_{11} & , & \ldots & , & a_{1m} \;) \in R \\ \vdots & & \vdots & & \vdots \\ (\; a_{n1} & , & \ldots & , & a_{nm} \;) \in R \end{array}}{(\; f(a_{11}, \ldots, a_{n1}) \; , \; \ldots \; , \; f(a_{1m}, \ldots, a_{nm}) \;) \in R}$$

Exemple

On considère la relation R suivante.

$$R = \{(0,0,0), (1,0,0), (0,0,1)\}$$

- R est close par l'opération binaire $\wedge$.
  Par exemple,

$$\begin{array}{ccccccc} & \wedge & & \wedge & & \wedge & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 0 & , & 0 & , & 0 &) \in R \end{array}$$

  (on peut vérifier les autres cas).
- Mais R n'est pas close par $\vee$ puisque,

$$\begin{array}{ccccccc} & \vee & & \vee & & \vee & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 1 & , & 0 & , & 1 &) \notin R \end{array}$$

Exemple

On considère la relation R suivante.

$$R = \{(0,0,0), (1,0,0), (0,0,1)\}$$

- R est close par l'opération binaire $\wedge$.
  Par exemple,

$$\begin{array}{ccccccc} & \wedge & & \wedge & & \wedge & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 0 & , & 0 & , & 0 &) \in R \end{array}$$

  (on peut vérifier les autres cas).
- Mais R n'est pas close par $\vee$ puisque,

$$\begin{array}{ccccccc} & \vee & & \vee & & \vee & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 1 & , & 0 & , & 1 &) \notin R \end{array}$$

Cas triviaux

- Un langage de contrainte booléen est de type 1-valide ssi
- Chaque relation de ce langage contient le tuple comportant que des 1
  ssi
- Chaque relation est close pour l'opération constante 1.

Dans ce cas, toute instance est trivialement satisfaite par l'assignement qui met à 1 toutes les variables. Complexité triviale $O(1)$.
Le cas 0-valide est similaire.

Horn

- Un langage de contrainte booléen est de type Horn
  ssi
- chaque relation R de $\mathcal{B}$ est close par l'opération booléenne
  $\wedge$
  ssi
- chaque relation R de $\mathcal{B}$ peut être simulée par une formule propositionnelle de Horn φ [4]:

$$t \in R \iff \mathcal{B} \models \varphi(t)$$

4. La réduction est polynomiale

Clôture des modèles de certaines clauses particulières

Exo : l'exemple des langages de type Horn

1. Soit C une clause de Horn à r variables. Soit R_C l'ensemble des assignements (un assignement est vu comme un tuple de r valeurs) t satisfaisant la clause C. Montrez que R_C est clos par $\wedge$: c'est-à-dire que pour tout t_1 et t_2 de R_C le tuple $t_1 \wedge t_2$ (on applique $\wedge$ composante par composante) est dans R_C.
2. Montrez comment simuler la relation $R = \{000, 100, 010, 011, 111\}$ par plusieurs clauses de Horn.
3. Montrez comment simuler une relation R close par $\wedge$ par plusieurs clauses de Horn.

Complexité Horn = polynomial

Lorsque le langage des contraintes est de type Horn, il est donc possible de réduire GENERALIZED SAT à HORN-SAT qui est dans P.

Nous allons voir une approche plus directe qui exploite explicitement la clôture par $\wedge$ pour obtenir une résolution polynomiale.

Complexité Horn = polynomial

Lorsque le langage des contraintes est de type Horn, il est donc possible de réduire GENERALIZED SAT à HORN-SAT qui est dans P.

Nous allons voir une approche plus directe qui exploite explicitement la clôture par $\wedge$ pour obtenir une résolution polynomiale.

Exo : GAC et Horn

Donnez un algorithme polynomial pour une restriction de GENERALIZED SAT à des relations de Horn

(indications: définir une généralisation de la notion de cohérence d'arc et utiliser la clôture par $\wedge$ pour guider le choix d'une solution particulière lorsque l'instance est cohérente).

Exo : GAC et Horn

Donnez un algorithme polynomial pour une restriction de GENERALIZED SAT à des relations de Horn

(indications: définir une généralisation de la notion de cohérence d'arc et utiliser la clôture par $\wedge$ pour guider le choix d'une solution particulière lorsque l'instance est cohérente).

Correction I

Chaque Variable x est équipée d'un domaine D_x. Une instance est 2-cohérente ssi $\forall x \forall d \in D_x$ et toute contrainte R portant (entre autres variables) sur x, il existe un tuple supportant la valeur d: c'est-à-dire qu'il existe un tuple t de R satisfaisant $t[x] = d$.

Il est clair que si ce n'était pas le cas, on pourrait enlever la valeur d du domaine D_x. C'est le principe de l'algo Gac

Correction II

GAC (GENERALIZED ARC-CONSISTENCY)
Tant que les domaines des variables ne sont pas stables
Pour une var x, une val $d \in D_x$, une contr R portant sur x
Si $\{t \in R$ tel que $t[x] = d\} = \emptyset$ alors enlever d de D_x

- Si GAC vide le domaine d'une instance quelconque, on peut répondre INSATISFAIT.
- Si ce n'est pas le cas – pour un langage de contrainte de type Horn – on peut répondre SATISFAIT.

Correction III

En effet, on peut démontrer que l'assignement t' consistant à choisir pour une variable x, la valeur minimale (pour $\wedge$) de D_x est une solution. Soit R une contrainte quelconque. Pour chaque variable x sur laquelle R porte, par cohérence il existe un tuple t de R tel que $t[x] = d$, où d est la valeur minimale de D_x. En prenant la conjonction de ces tuples, on obtient un tuple de R (puisque le langage est de type Horn) compatible avec l'assignement t'.

Dual Horn

- Un langage de contrainte booléen est de type dual Horn ssi
- chaque relation R de $\mathcal{B}$ est close par l'opération booléenne $\bigvee$
  ssi
- chaque relation R de $\mathcal{B}$ peut être simulée par une formule propositionnelle dual-Horn φ [5]:

$$t \in R \iff \mathcal{B} \models \varphi(t)$$

Cas dual du précédent.

5. La réduction est polynomiale

Bijonctif

- Un langage de contrainte booléen est de type bijonctif
  ssi
- chaque relation R de $\mathcal{B}$ est close par l'opération booléenne majorité m
  ssi
- chaque relation R de $\mathcal{B}$ peut être simulée par une formule propositionnelle 2-Sat φ [6]:

$$t \in R \iff \mathcal{B} \models \varphi(t)$$

6. La réduction est polynomiale

Complexité Bijonctif = Polynomial

Lorsque le langage des contraintes est bijonctif, il est possible de réduire GENERALIZED SAT à 2-SAT (qui est dans P, cf. exos).

Une approche plus directe qui exploite explicitement la clôture par l'opération de majorité m permet d'obtenir une résolution polynomiale (voir exo du script).

Exo : Cas bijonctif

1. Soit C une 2-clause et R_C la relation binaire des assignements satisfaisant cette clause C. Montrez que R_C est close par l'opération de majorité m. Cette opération ternaire retourne l'argument le plus fréquent: e.g. $m(0,0,0) = m(0,0,1) = m(1,0,0) = m(0,1,0) = 0$.
2. Soit R une relation close par l'opération m. Réduisez la restriction de GENERALIZED SAT au langage de contrainte $\{R\}$ à 2-Sat (indication: remplacer la contrainte R par toutes les contraintes induites par les projections sur deux coordonnées).

Cas affine

- Un langage de contrainte booléen est de type affine
  ssi
- chaque relation R de $\mathcal{B}$ est close par l'opération booléenne ternaire $l(x, y, z) = x \oplus y \oplus z$
  ssi
- une instance peut être simulée par un système linéaire (où l'addition est modulo 2).

Complexité affine = Polynomial

- Relation = ensemble solution d'un système linéaire
- extraction d'une base
- système correspondant à la relation
- Système combinant les sous-systèmes pour chaque relation
- Résolution en temps polynomial (élimination de Gauss)

Récapitulatif des cas polynomiaux

- 0-valide, 1-valide (triviaux) c_0, c_1
- Horn et Dual-Horn (generalized arc consistency) $\wedge, \vee$
- bijonctif (path-consistency) m
- affine (élimination de Gauss) l

Proposition

Soit $\mathcal{B}$ une structure booléenne. Si les relations de $\mathcal{B}$ sont closes par $c_0, c_1, \wedge, \vee, m$ ou l alors CSP($\mathcal{B}$) est dans P.

Troisième partie III

Approche algébrique et théorème de Schaefer

Plan du cours

Expressivité et p.p. définissabilité

Théorie des clones

Nouvelles relations à partir d'anciennes

- Considérez les deux relations suivantes:

$$\begin{aligned} R_0 &= \{(a,b,c) \in \{0,1\}^3 \mid a+b+c = 0 \pmod 2\} \\ R_1 &= \{(a,b,c) \in \{0,1\}^3 \mid a+b+c = 1 \pmod 2\} \end{aligned}$$

- À quel point le langage de contraintes $\{R_0, R_1\}$ est-il expressif?
- Quelles autres relations peut-on "simuler" en n'utilisant que R_0 et R_1?

Nouvelles relations à partir d'anciennes

Quelles autres relations peut-on "simuler" en n'utilisant que R_0 et R_1 ?

$$\begin{aligned} R_0 &= \{(a,b,c) \in \{0,1\}^3 \mid a+b+c = 0 \pmod 2\} \\ R_1 &= \{(a,b,c) \in \{0,1\}^3 \mid a+b+c = 1 \pmod 2\} \end{aligned}$$

$$R_1(x,x,x) \qquad x = 1$$

$$R_0(x,x,x) \qquad x = 0$$

$$z = 0, R_0(x,y,z) \qquad x + y = 0$$

(z est une nouvelle variable)

$$x + y + v = 0, v + z + w = 0 \qquad x + y + z + w = 0$$

(v est une nouvelle variable)

Nouvelles relations à partir d'anciennes

- Soit R les solutions d'une équation linéaire arbitraire modulo 2:

$$R = \{(x_1, x_2, \ldots, x_k) \in \{0,1\}^k \mid x_1 + x_2 + \cdots + x_k = b\},$$

où $b = 0$ ou $b = 1$.

- On peut alors "simuler" (ou exprimer) R en utilisant seulement R_0 et R_1.

$$\begin{array}{ccc}
R_0(z,z,z) & z+z+z=0 & z=0 \\
R_0(z,x_1,v_1) & z+x_1+v_1=0 & v_1=x_1 \\
R_0(v_1,x_2,v_2) & v_1+x_2+v_2=0 & v_2=x_1+x_2 \\
\vdots & \vdots & \vdots \\
R_0(v_{k-1},x_{k-1},v_k) & v_{k-1}+x_{k-1}+v_k=0 & v_k=x_1+\cdots+x_{k-1} \\
R_b(v_k,x_k,z) & v_k+x_k+z=b & x_1+\cdots+x_k=b
\end{array}$$

- Ainsi, on peut exprimer n'importe quelle relation affine en utilisant seulement R_0 et R_1.

Exos : expressivité d'un langage (début)

1. Simulez la relation booléenne $x \neq y$ par des 2-clauses. En déduire une réduction de 2-col à 2-Sat.
2. Donnez une réduction de K_5-Col à C_5-Col. Quelle est la complexité de C_5-Col?

Expressivité et p.p. définissabilité

Théorie des clones

Expressivité d'un langage de contraintes

Soit Γ l'ensemble des relations d'une structure booléenne $\mathcal{B}$.
Intuition: Plus Γ est riche (expressif), plus le problème associé est difficile à résoudre.

Exemple

Si Γ contient deux relations binaire R_1 and R_2. Considérons l'instance

$$((x,z),R_1),\ ((z,y),R_2).$$

- La contrainte implicite sur (x,y) est $R_3 = R_1 \circ R_2$
- $R_3(x,y) = \exists z R_1(x,z) \wedge R(z,y)$.
- $R3 \notin \Gamma$, mais
- Les langages Γ et $\Gamma \cup \{R_3\}$ sont équivalents (du point de vue de l'expressivité) et de plus les problèmes de contraintes correspondant sont équivalents (inter-réductibles en temps polynomial).

Expressivité d'un langage de contraintes

Soit Γ l'ensemble des relations d'une structure booléenne $\mathcal{B}$.
Intuition: Plus Γ est riche (expressif), plus le problème associé est difficile à résoudre.

Exemple

Si Γ contient deux relations binaire R_1 and R_2. Considérons l'instance

$$((x,z),R_1),\ ((z,y),R_2).$$

- La contrainte implicite sur (x,y) est $R_3 = R_1 \circ R_2$
- $R_3(x,y) = \exists z R_1(x,z) \wedge R(z,y)$.
- $R3 \notin \Gamma$, mais
- Les langages Γ et $\Gamma \cup \{R_3\}$ sont équivalents (du point de vue de l'expressivité) et de plus les problèmes de contraintes correspondant sont équivalents (inter-réductibles en temps polynomial).

Expressivité (formellement)

On note par $[\Gamma]$ l'ensemble des relations qui peuvent être simulées par les relations de Γ.
Combinatoirement, il s'agit de toutes les relations qu'on peut construire en fabriquant des $\ll$ gadgets $\gg$à l'aide de variable annexes et de relations de Γ.
Plus formellement, il s'agit de toutes les relations qui sont interprétables par une formule primitive positive sur Γ, c'est-à-dire utilisant

- des relations de $\Gamma \cup \{=_D\}$,
- la conjonction $\bigwedge$, et
- la quantification existentielle $\exists$.

Exos : expressivité d'un langage (fin)

- Exprimez la réduction de K_5 à C_5 comme une p.p. réduction. On rappelle qu'une *p.p. réduction* est déterminée par une formule du premier-ordre utilisant les symboles de relations de Γ, la conjonction $\wedge$, la quantification existentielle $\exists$ et l'égalité $=$.

Expressivité et p.p. définissabilité

Théorie des clones

Clones relationnels

Définition

On note $[\Gamma]$ l'ensemble des relations qui peuvent être simulées par les relations de Γ. Il s'agit de toutes les relations qui sont interprétables par une formule primitive positive sur Γ, c'est-à-dire utilisant

- des relations de $\Gamma \cup \{=_D\}$,
- la conjonction $\wedge$, et
- la quantification existentielle $\exists$.

$[\Gamma]$ est appelé le clone relationnel généré par Γ.

Théorème (Jeavons '98)

Si Γ_1 *et* Γ_2 *sont des langages de contraintes tels que* $[\Gamma_1] \subseteq [\Gamma_2]$ *alors* $\mathsf{SAT}(\Gamma_1)$ *se (logspace-)réduit à* $\mathsf{SAT}(\Gamma_2)$.

Slogan: $[\Gamma]$ capture l'expressivité de Γ.

Invariances et polymorphismes

Définition

Soit Γ un ensemble de relations booléennes. Une fonction booléenne f d'arité n préserve les relations de Γ ssi pour toute relation R de Γ et tous tuples $t_1, t_2, \ldots, t_n$ appartenant à cette relation R, le tuple $f(t_1, t_2, \ldots, t_r)$ (on applique f composante par composante) appartient aussi à R.
On dit aussi que f est un polymorphisme de Γ ou encore que les relations R sont invariantes par f.

Invariances et polymorphismes

Définition

Soit Γ un ensemble de relations booléennes. Une fonction booléenne f d'arité n préserve les relations de Γ ssi pour toute relation R de Γ et tous tuples $t_1, t_2, \ldots, t_n$ appartenant à cette relation R, le tuple $f(t_1, t_2, \ldots, t_r)$ (on applique f composante par composante) appartient aussi à R.
On dit aussi que f est un polymorphisme de Γ ou encore que les relations R sont invariantes par f.

Application composante par composante

Une fonction booléenne f d'arité n préserve une relation R d'arité m ssi pour tous tuples $t_1 = (a_{11}, \ldots, a_{1m}) \ldots t_n = (a_{n1}, \ldots, a_{nm})$ appartenant à cette relation R, le tuple $f(t_1, \ldots, t_r)$ (on applique f composante par composante) appartient aussi à R.

$$\frac{\begin{array}{ccccccc} & f & & f & & f & \\ (& a_{11} & , & \ldots & , & a_{1m} &) \in R \\ & \vdots & & \vdots & & \vdots & \vdots \\ (& a_{n1} & , & \ldots & , & a_{nm} &) \in R \end{array}}{\begin{array}{ccccc} (& f(a_{11}, \ldots, a_{n1}) & , & \ldots & , & f(a_{1m}, \ldots, a_{nm}) &) \in R \end{array}}$$

Exemple

On considère la relation R suivante.

$$R = \{(0,0,0), (1,0,0), (0,0,1)\}$$

- l'opération binaire $\wedge$ un polymorphisme de R.
  Par exemple,

$$\begin{array}{ccccccc} & \wedge & & \wedge & & \wedge & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 0 & , & 0 & , & 0 &) \in R \end{array}$$

  (on peut vérifier les autres cas).
- l'opération binaire $\vee$ n'est pas un polymorphisme de R puisque,

$$\begin{array}{ccccccc} & \vee & & \vee & & \vee & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 1 & , & 0 & , & 1 &) \notin R \end{array}$$

Exemple

On considère la relation R suivante.

$$R = \{(0,0,0), (1,0,0), (0,0,1)\}$$

- l'opération binaire $\wedge$ un polymorphisme de R.
  Par exemple,

$$\begin{array}{ccccccc} & \wedge & & \wedge & & \wedge & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 0 & , & 0 & , & 0 &) \in R \end{array}$$

  (on peut vérifier les autres cas).
- l'opération binaire $\vee$ n'est pas un polymorphisme de R puisque,

$$\begin{array}{ccccccc} & \vee & & \vee & & \vee & \\ (& 1 & , & 0 & , & 0 &) \in R \\ (& 0 & , & 0 & , & 1 &) \in R \\ \hline (& 1 & , & 0 & , & 1 &) \notin R \end{array}$$

Clones d'opérations

Soit F un ensemble d'opérations.
Soit $\langle F \rangle$ l'ensemble des opérations obtenues par composition des opérations de F (et des projections).

E.g. si $f_1, f_2 \in F$ alors $f_2(f_1(x,y), x, f_2(f_1(z,y), x, z)) \in \langle F \rangle$.

vocabulaire
$\langle F \rangle$ est le clone généré par F.

Correspondance de Galois

Notation

Soit Γ un ensemble de relations booléennes et F un ensemble de fonctions booléennes.

- $Pol(\Gamma)$ dénote l'ensemble des polymorphismes de Γ.
- $Inv(F)$ dénote l'ensemble des relations invariantes par les fonctions de F.

Théorème (Geiger '68; Bodnarchuk et al. '69)

- *Pour tout langage de contrainte* Γ, $[\Gamma] = Inv\big(Pol(\Gamma)\big)$
- *Pour tout ensemble d'opérations* F, $\langle F \rangle = Pol\big(Inv(F)\big)$

Slogan: les polymorphismes contrôlent le pouvoir d'expression.

Correspondance de Galois

Notation

Soit Γ un ensemble de relations booléennes et F un ensemble de fonctions booléennes.

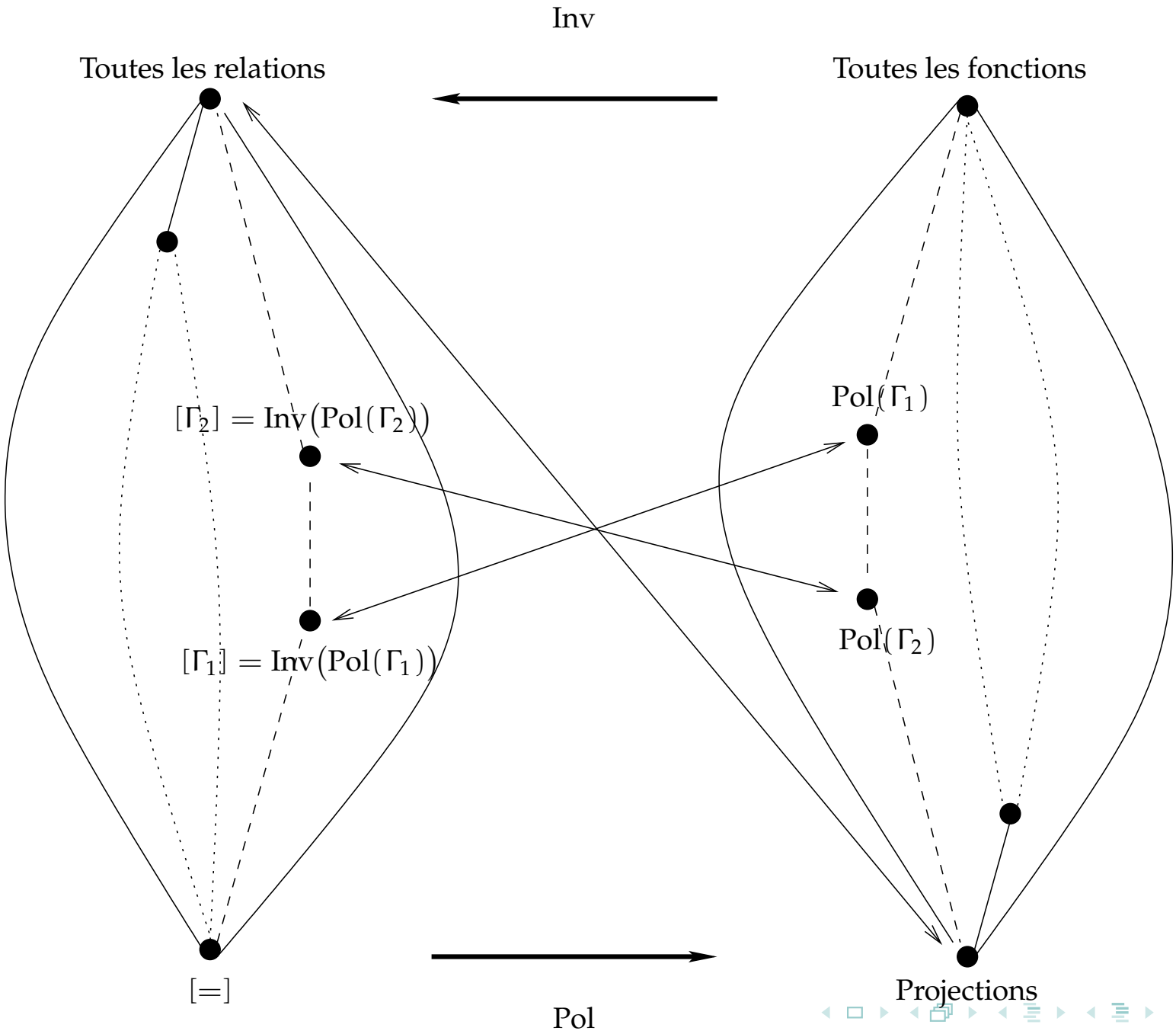
- $Pol(\Gamma)$ dénote l'ensemble des polymorphismes de Γ.
- $Inv(F)$ dénote l'ensemble des relations invariantes par les fonctions de F.

Théorème (Geiger '68; Bodnarchuk et al. '69)

- *Pour tout langage de contrainte* Γ, $[\Gamma] = Inv\big(Pol(\Gamma)\big)$
- *Pour tout ensemble d'opérations* F, $\langle F \rangle = Pol\big(Inv(F)\big)$

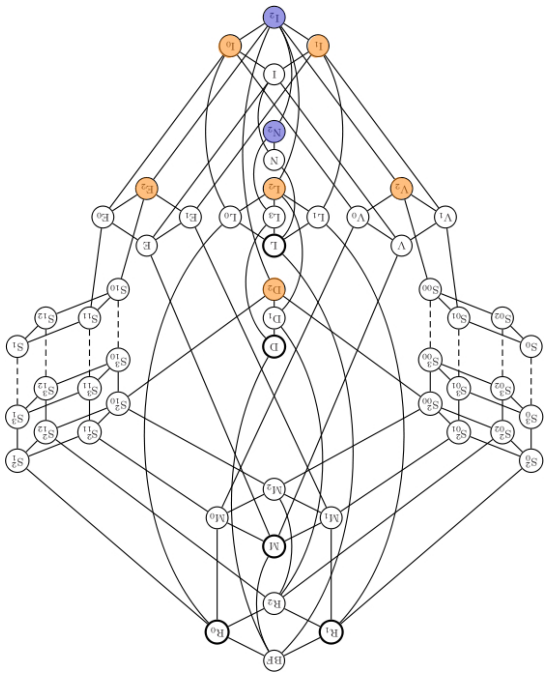
Slogan: les polymorphismes contrôlent le pouvoir d'expression.

Correspondance de Galois

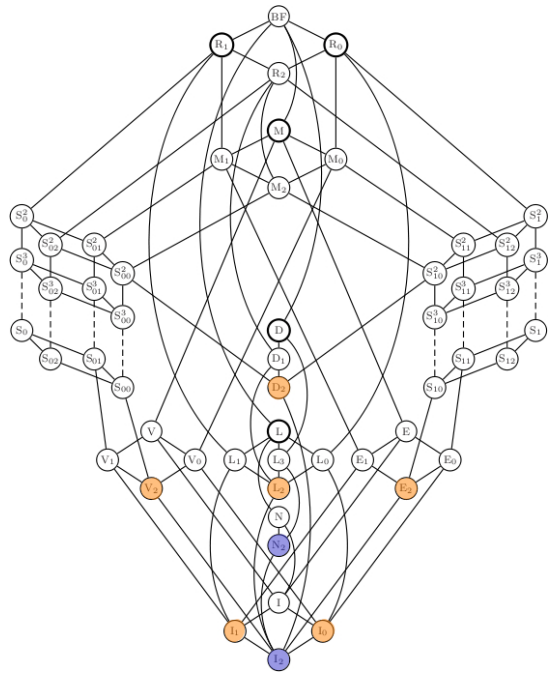


Correspondance de Galois

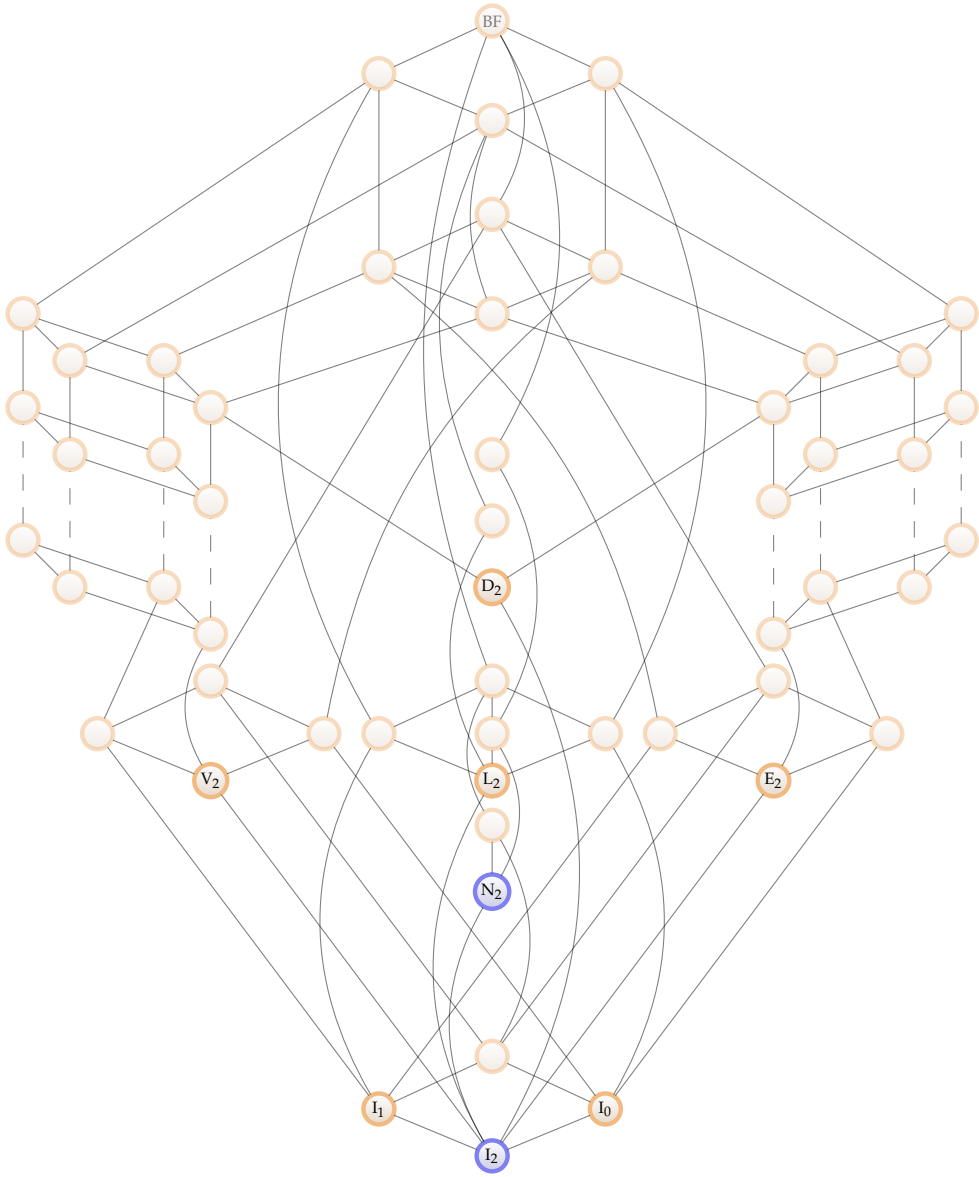
Clones relationnels



clones fonctionnels



Treillis de Post



Quatrième partie IV

Les cas NP-complets du théorème de Schaefer

Plan du cours

Fin de la preuve

Conclusion

Récapitulatif des cas polynomiaux

- 0-valide, 1-valide (triviaux) c_0, c_1
- Horn et Dual-Horn (generalized arc consistency) $\wedge, \vee$
- bijonctif (path-consistency) m
- affine (élimination de Gauss) l

Proposition

Soit $\mathcal{B}$ une structure booléenne. Si les relations de $\mathcal{B}$ sont closes par $c_0, c_1, \wedge, \vee, m$ ou l alors CSP($\mathcal{B}$) est dans P.

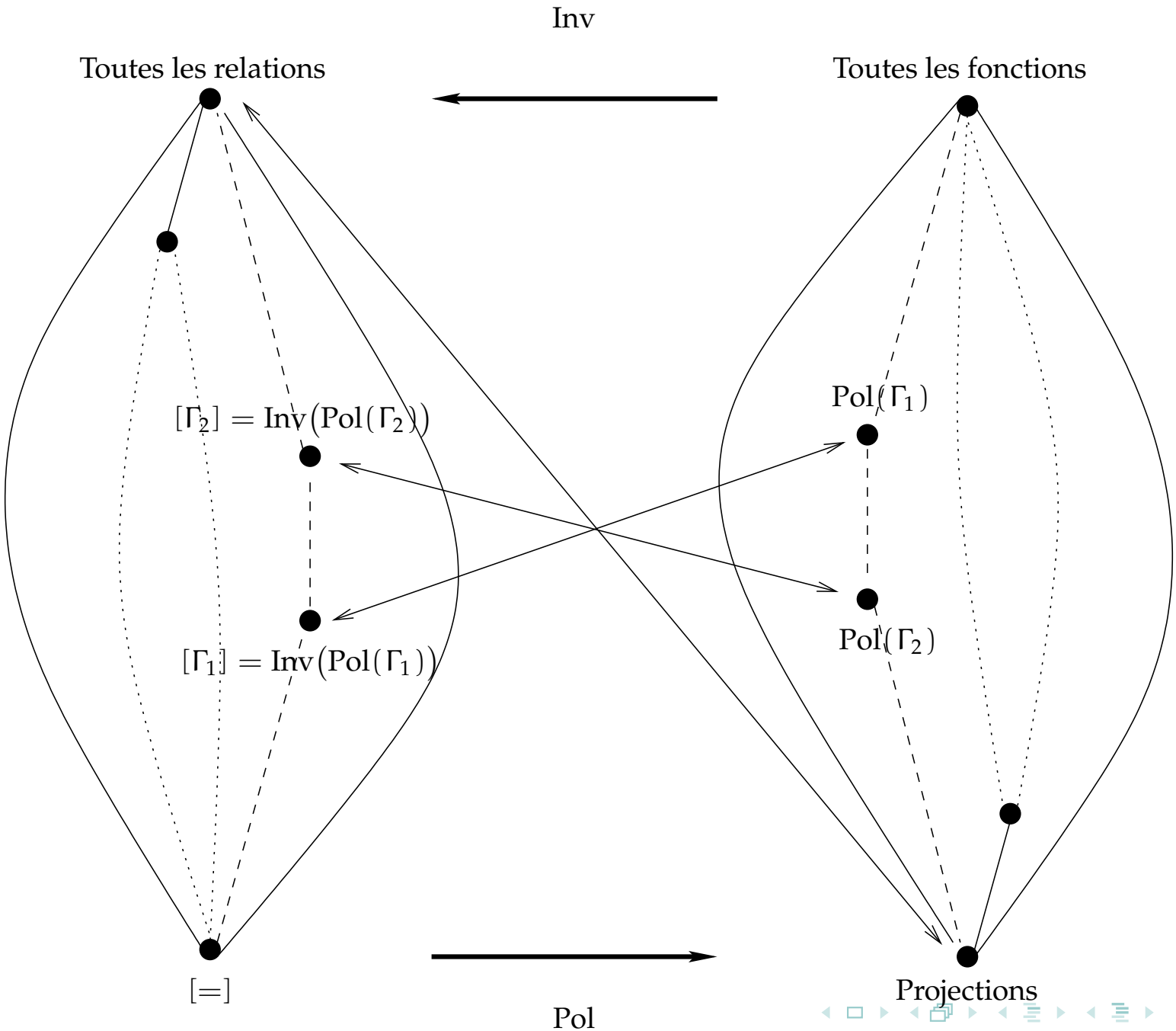
Expressivité

- $[\Gamma]$ expressivité du langage de contrainte Γ (ensemble des relations p.p.-définissables).
- Correspondance de Galois: $[\Gamma] = Inv\big(Pol(\Gamma)\big)$

Théorème (Jeavons '98)

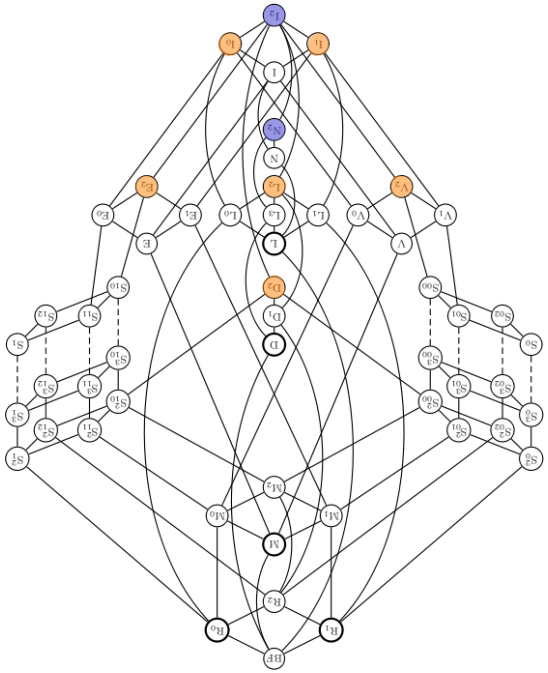
Si Γ_1 *et* Γ_2 *sont des langages de contraintes tels que* $[\Gamma_1] \subseteq [\Gamma_2]$ *alors* $\mathsf{SAT}(\Gamma_1)$ *se réduit à* $\mathsf{SAT}(\Gamma_2)$*.*

Correspondance de Galois

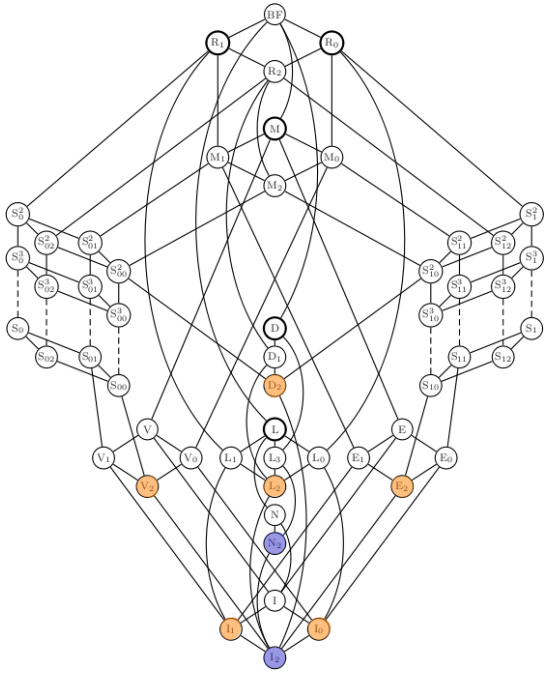


Correspondance de Galois

Clones relationnels



clones fonctionnels



Fin de la preuve

Conclusion

Cas NP-complets

Pour terminer la preuve du théorème de Schaefer, il nous reste à montrer que

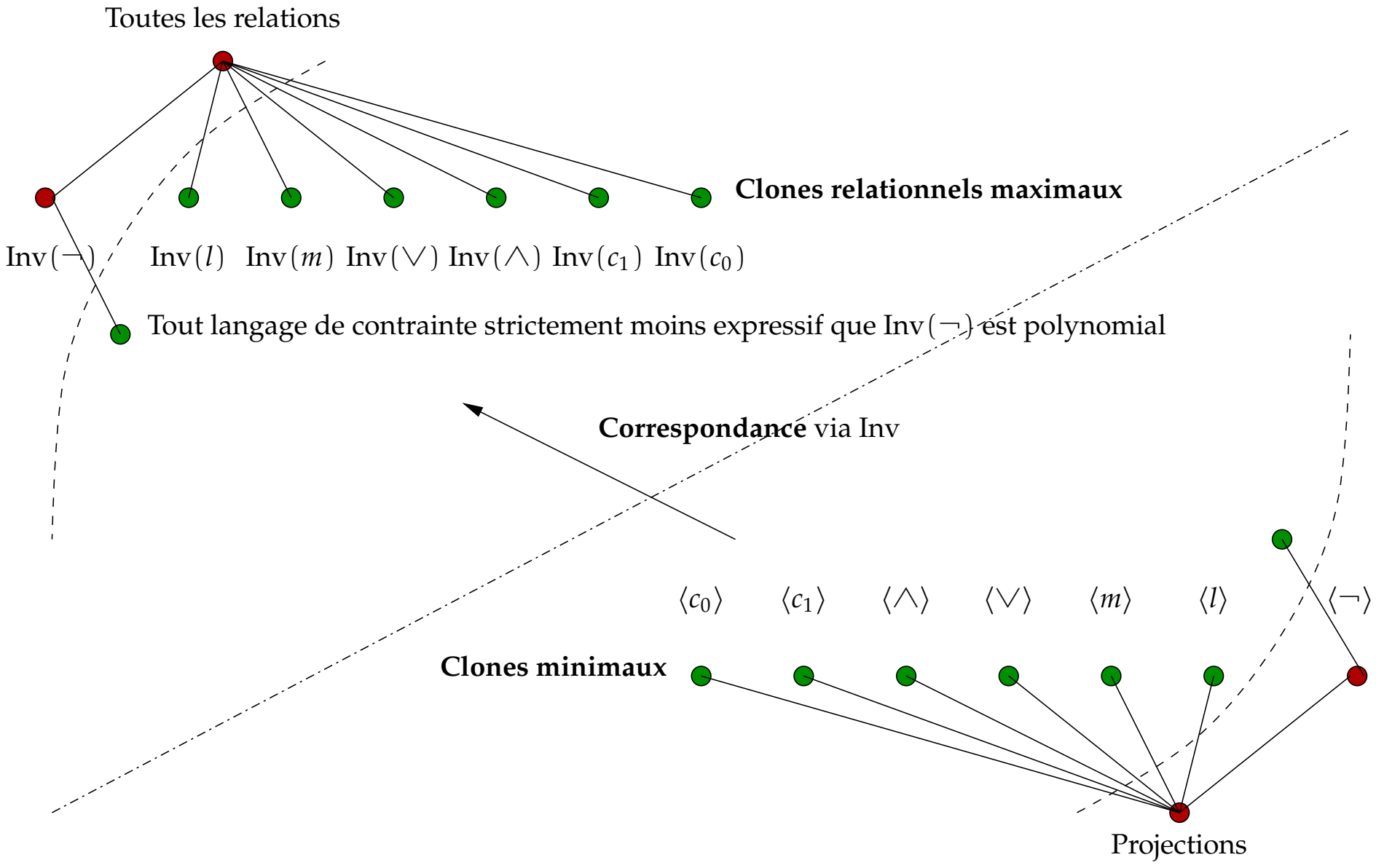
Proposition

Si Γ n'est pas clos par l'une des 6 opérations suivantes $c_0, c_1, \wedge, \vee, l, m$, alors SAT*($\Gamma$) est NP-complet.*

Preuve

Si Γ n'est pas un cas polynomial alors $Pol(\Gamma)$ ne contient pas l'une des 6 opérations suivantes $c_0, c_1, \wedge, \vee, l, m$.
Donc $\mathrm{Pol}(\Gamma)$ est soit le clone $N_2 = \langle \neg \rangle$, soit le clone trivial I_2 (toutes les projections).
Donc soit $[\Gamma] = \mathrm{Inv}(\mathrm{Pol}(\Gamma)) = \mathrm{Inv}(\{\neg\})$ qui contient la relation R_{NAE} de NOT-ALL-EQUAL-3-SAT, ou bien $[\Gamma]$ est l'ensemble de toutes les relations (qui contient aussi la relation R_{NAE}).
Dans tous les cas, d'après le théorème de Jeavons, on a SAT(R_{NAE}) qui se réduit à SAT(Γ). On va voir que SAT(R_{NAE}), plus connu sous le nom de NOT-ALL-EQUAL-3-SAT, est NP-complet. Donc SAT(Γ) est NP-complet.

Illustration de la preuve



NOT-ALL-EQUAL-3-SAT est NP-complet

- On verra plus tard pourquoi mais il n'est pas possible de réduire 3-Sat à ce problème en faisant de la "simulation par gadget".
- On fait une réduction en deux étapes:
  - de 3-SAT à NOT-ALL-EQUAL-4-SAT
  - puis à son cousin NOT-ALL-EQUAL-3-SAT

Détail première étape

- on se donne une variable spéciale θ pour toute la formule
- chaque variable x_i de 3-SAT devient une variable y_i pour NOT-ALL-EQUAL-4-SAT
- une 3-clause $x_i \vee x_j \vee x_k$ de 3-SAT devient une 4-clause (y_i, y_j, y_k, θ)
- la correspondance entre assignement est lue par: x_i est vrai ssi y_i n'a pas la même valeur que θ

Détail première étape

- on se donne une variable spéciale θ pour toute la formule
- chaque variable x_i de 3-SAT devient une variable y_i pour NOT-ALL-EQUAL-4-SAT
- une 3-clause $x_i \vee x_j \vee x_k$ de 3-SAT devient une 4-clause (y_i, y_j, y_k, θ)
- la correspondance entre assignement est lue par:
  x_i est vrai ssi y_i n'a pas la même valeur que θ

Détail seconde étape

- Nous procédons de manière similaire à la réduction k-SAT vers 3-SAT en ajoutant des variables supplémentaires.
- Une clause (x_1, x_2, x_3, x_4) devient $(x_1, x_2, w) \wedge (\neg w, x_3, x_4)$.

J'ai un peu triché: dans la définition de NOT-ALL-EQUAL-3-SAT comme problème de contrainte, nous n'avons pas le droit au littéraux! On peut pallier facilement à ce problème en remarquant que la contrainte NOT-ALL-EQUAL-3-SAT sur (x, x, x') impose $x = \neg x'$.

Détail seconde étape

- Nous procédons de manière similaire à la réduction k-SAT vers 3-SAT en ajoutant des variables supplémentaires.
- Une clause (x_1, x_2, x_3, x_4) devient $(x_1, x_2, w) \wedge (\neg w, x_3, x_4)$.

J'ai un peu triché: dans la définition de NOT-ALL-EQUAL-3-SAT comme problème de contrainte, nous n'avons pas le droit au littéraux! On peut pallier facilement à ce problème en remarquant que la contrainte NOT-ALL-EQUAL-3-SAT sur (x, x, x') impose $x = \neg x'$.

Cas NP-complets

Nous avons donc démontré la proposition suivante.

Proposition

Si Γ n'est pas clos par l'une des 6 opérations suivantes $c_0, c_1, \wedge, \vee, l, m$, alors SAT*($\Gamma$) est NP-complet.*

Théorème de Schaefer comme corollaire de la classification de Post

Théorème (Post, 1941)

Pour tout langage de contrainte Γ *sur* $D = \{0, 1\}$*, on est dans l'un des cas suivants*

- *Les constantes 0 ou 1 sont dans* $\mathrm{Pol}(\Gamma)$ — $\textsc{Sat}(\Gamma)$ *est trivial*
- $\wedge \in \mathrm{Pol}(\Gamma)$ — $\textsc{Sat}(\Gamma) \subseteq$ ***Horn-Sat***
- $\vee \in \mathrm{Pol}(\Gamma)$ — $\textsc{Sat}(\Gamma) \subseteq$ ***dual Horn-Sat***
- $m \in \mathrm{Pol}(\Gamma)$ — $\textsc{Sat}(\Gamma) \subseteq$ ***2-Sat***
- l *est dans* $\mathrm{Pol}(\Gamma)$ — $\textsc{Sat}(\Gamma) \subseteq$ ***équations linéaires***
- $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}\langle R_{NAE} \rangle$. — ***NAE-Sat*** $\subseteq \textsc{Sat}(\Gamma)$

Corollaire (Schaefer, 1978)

Pour tout langage de contrainte Γ *sur* $D = \{0, 1\}$*,* $\textsc{Sat}(\Gamma)$ *est soit polynomial soit NP-complet.*

Uniformisation

En analysant notre preuve du théorème de Schaefer, nous pouvons remarquer que la preuve fonctionne pour des langages de contraintes arbitraires (pas forcément finis). En fait ceci est dû au fait que les classes polynomiales uniformisent: par exemple, l'algorithme de résolution de $\textsc{Sat}(\Gamma)$ où Γ est un langage de contrainte fini de type Horn est le même et s'applique au cas où Γ est l'ensemble non fini de toutes les relations de Horn.

Méta-problème et algorithme adaptatif

Méta-problème

On peut détecter en temps polynomial si les relations d'une instance forme un langage de contraintes polynomial puisqu'il suffit de tester si ces relations sont closes pour l'une des 6 opérations booléennes $c_0, c_1, \wedge, \vee, m$ ou l (pour une relation la complexité est $O(e^3 m)$, où e est le nombre de tuples et m l'arité puisqu'on teste au pire une fonction ternaire nécessitant de vérifier $\binom{e}{3} = O(e^3)$ possibilités).

Algorithme adaptatif

- Si les relations de l'instance appartiennent à un langage de contraintes polynomial, on le détecte en temps polynomial (puisque le méta-problème est polynomial) puis on appelle l'algorithme de résolution polynomial associé.
- Sinon on utilise la méthode générale Search + propagation.

Exo : I

1. Pour chacune des relations suivantes, indiquez si elles sont closes ou non pour chacune des opération suivantes c_0, c_1 (opérations constantes), $\wedge, \vee, l$ (opération ternaire: somme modulo 2 de ces arguments), m (majorité ternaire) et $\neg$ (unaire, complément).

$$R_1 = \{000, 100, 010, 011, 111\} \quad R_2 = \{000, 111\}$$
$$R_3 = \{01, 11\} \quad R_4 = \{01, 10\}$$
$$R_5 = \{100, 010, 001\} \quad R_6 = \{001, 010, 011, 100, 101, 110\}$$

2. Quelle est la complexité de GENERALIZED SAT pour les langages suivants?

$$\Gamma_1 = \{R_1, R_2, R_3\} \quad \Gamma_2 = \{R_1, R_4\}$$
$$\Gamma_3 = \{R_2, R_4\} \quad \Gamma_4 = \{R_3, R_6\}$$

Exo : II

3. Montrez que pour tous les cas polynomiaux du théorème de Schaefer, on peut également trouver une solution en temps polynomial si elle existe (indication: on utilisera les algos du théorème de Schaefer comme des boîtes noires).

Fin de la preuve

Conclusion

Conclusion

Autres questions pour 2 valeurs

- Le treillis de Post permet de classifier la complexité pour d'autres questions : circomscription, abduction, comptage, énumération
- mais pas toujours : MaxSat, approximation.

Plus de 2 valeurs

- L'approche algébrique s'étant mais ne permet pas de classification complète puisque le treillis des clones est non dénombrable et largement inconnu.
- Il faut encore plus d'outils algébriques: classification d'algèbres en analysant le type local (*congruence tame theory*).
- Bulatov démontre ainsi une dichotomie pour 3 valeurs et pour les langages dit conservatifs (qui contiennent toutes les contraintes unaires).

Pour démontrer la conjecture

Conjecture (dernière forme connue)

Soit Γ un langage de contraintes quelconque sur un domaine fini. Si $\mathrm{Pol}(\Gamma)$ contient un *terme de Siggers* alors $\textsc{Csp}(\Gamma)$ est polynomial.

Un terme de Siggers est une opération d'arité 4 satisfaisant les identités $f(x,x,x,x) = x$ et $f(y,x,y,z) = f(x,y,z,x)$.

Il suffit de traiter le cas de digraphes.

Pour démontrer la conjecture

Conjecture (dernière forme connue)

Soit Γ un langage de contraintes quelconque sur un domaine fini. Si $\mathrm{Pol}(\Gamma)$ contient un *terme de Siggers* alors $\textsc{Csp}(\Gamma)$ est polynomial.

Un terme de Siggers est une opération d'arité 4 satisfaisant les identités $f(x,x,x,x) = x$ et $f(y,x,y,z) = f(x,y,z,x)$.

Il suffit de traiter le cas de digraphes.

Bibliographie

- pour une biblio plus complète et une liste de résumés récents, voir le méta-résumé (sur ma page web)
  Florent Madelaine. *De la complexité des problèmes de contraintes*. JFPC'2011.
- pour une extension de la méthode algébrique aux problèmes de contraintes quantifiés
  Hubie Chen. *A Rendezvous of Logic, Complexity, and Algebra*. ACM Computing Surveys, Volume 42, Issue 1, December 2009.

Cinquième partie V

Solveurs modernes

Plan du cours

TO DO