Sur la génération de groupes finis non isomorphes

Gilles Audemard
LIM,
39, Rue Joliot-Curie
13453 Marseille cedex 13
audemard@lim.univ-mrs.fr

Laurent Henocque
LIM,
39, Rue Joliot-Curie
13453 Marseille cedex 13
henocque@lim.univ-mrs.fr

Résumé

Nous présentons un algorithme permettant de générer des modèles finis de théories du premier ordre. Cet algorithme s'avère très performant sur une catégorie particulière des théories équationnelles, les groupes abéliens, qui résistent aux approches connues. Pendant la recherche, et sans surcoût de temps, la plupart des interprétations isomorphes sont écartées. Cela permet de n'explorer qu'un nombre très limité de noeuds. L'idée principale de l'article est d'exploiter l'existence, dans la formulation du problème, d'une fonction unaire bijective (l'inverse de la loi commutative dans le cas d'un groupe abélien) et de focaliser la recherche sur celle ci en générant de façon constructive ses seules interprétations canoniques. Cette fonction étant bijective, de nombreux sous espaces isomorphes sont identifiés et supprimés pendant la suite de la recherche. Cet algorithme est implanté sur le générateur de modèles SEM. A notre connaissance, c'est la première fois qu'un algorithme génère tous les groupes abéliens d'ordre 63.

1 Introduction

Les théories équationnelles présentent un grand nombre de problèmes difficiles. Dans [10], Jian Zhang définit un ensemble de problèmes qui peuvent former un challenge pour les générateurs de modèles finis de théories équationnelles. Ces problèmes ont été résolus par différentes approches : FALCON [9], SEM [11], MGTP-G [6], LDPP, SATO [8], MACE [4] et FMC [5]. Les groupes abéliens finis constituent un problème difficile ayant résisté à de nombreuses approches. Récemment, diverses évolutions de SEM ont été implantées ([1], [2], [3]) et n'ont pas permis d'améliorer les performances sur ces instances, malgré des résultats significatifs par ailleurs.

Une théorie équationnelle est constituée d'un ensemble d'axiomes : des formules de la logique du premier ordre utilisant uniquement le prédicat d'égalité ($\forall x, \forall y, \forall z : h(f(x,y)) = f(z,x)$ par exemple). Dans cet article, nous ne considérons que des théories où toutes les variables sont universellement quantifiées. Trouver un modèle d'une telle théorie revient à donner une interprétation aux symboles fonctionnels sur un domaine D_n qui satisfasse les axiomes. L'existence d'un modèle prouve la consistance d'une théorie. L'existence d'un contre modèle peut servir à réfuter une conjecture.

Le problème de la génération de modèles finis pour les théories équationnelles peut être vu comme un problème de satisfaction de contraintes particulier (CSP), où les contraintes sont très symétriques. Ces nombreuses symétries proviennent notamment de la quantification universelle de toutes les variables. Les différents générateurs de modèles existant dans la littérature utilisent différentes approches pour éliminer des interprétations isomorphes. MGTP-G [6] ajoute des axiomes en début de recherche afin de supprimer statiquement certains isomorphismes. Quand elle peut être utilisée, cette technique est l'une des plus efficaces. FALCON [9] et SEM [11] utilisent un filtrage dynamique et l'heuristique LNH ("Least Number Heuristic") pour éviter d'explorer des sous espaces isomorphes. L'idée de LNH est d'éviter de parcourir des branches correspondant à des valeurs connues comme interchangeables car elles n'ont pas encore été utilisées pendant la recherche. Il suffit pour cela de maintenir à jour une marque, que l'on appelle *mdn* ("Max Designated Number"), égal au plus grand élément du domaine référencé par un point de choix. Cette approche est très efficace car elle ne génère aucun surcoût de temps de calcul. Elle a permis de résoudre quelques problèmes ouverts.

Nous proposons de généraliser l'heuristique LNH afin de supprimer un plus grand nombre d'interprétations isomorphes. Cette nouvelle heuristique, que nous appelons XLNH (pour eXtended LNH), a donné des résultats très intéressants pour la génération de modèles finis de groupes commutatifs.

Cet article est organisé comme suit : La section 2 définit les théories équationnelles. La section 3 décrit la proposition d'équivalence de modèles. La procédure d'énumération est décrite à la section 4. Des résultats expérimentaux sont proposés à la section 5. La section 6 conclut.

2 Théories équationnelles

2.1 Syntaxe

Nous utilisons un sous ensemble $\mathcal{L}$ de la logique du premier ordre, sans quantificateurs existentiels, avec l'égalité comme seul prédicat. Dans $\mathcal{L}$ toutes les variables sont quantifiées universellement. Le symbole d'inégalité ($\neq$) dénote la négation de l'égalité. L'ensemble des variables est $\{x,y,z,x_1,\dots\}$. Les constantes sont des entiers de l'ensemble $\{0,1,2,\dots\}$. Un symbole fonctionnel peut être identifié de n'importe qu'elle manière, dès qu'il n'y a pas d'ambiguïté avec les catégories précédentes. Un terme est construit récursivement à partir des symboles fonctionnels, des variables et des constantes.

Comme toutes les variables sont universellement quantifiées, les quantificateurs sont généralement omis. La figure 1 illustre les possibilités offertes par ce langage. $\mathcal{L}$ peut décrire les axiomes de problèmes mathématiques comme les groupes abéliens ou les anneaux unitaires. Comme $\mathcal{L}$ ne comporte que le prédicat d'égalité, les problèmes issus de $\mathcal{L}$ sont souvent appelés théories équationnelles. Il peut être intéressant pour les mathématiciens de prouver ou de réfuter l'existence de structures finies satisfaisant certains axiomes. On peut noter que les théories équationnelles forment le plus simple langage sur lequel nos résultats s'appliquent. Les concepts introduits dans cet article peuvent être étendus à des langages plus riches comme, par exemple, les théories multi sortes utilisés par le générateur de modèles finis SEM [11]. C'est une voie pour de futures recherches.

$$h(x,0) = x$$
$$h(0,x) = x$$
$$h(x,g(x)) = 0$$
$$h(g(x),x) = 0$$
$$h(h(x,y),z) = h(x,h(y,z))$$
$$h(x,y) = h(y,x)$$

FIG. 1 – *Axiomes du groupe abélien*

2.2 Sémantique

Sans perte de généralité, les individus sont représentés par l'ensemble $N = \{0,1,2\ldots\}$ des entiers naturels. Comme nous nous intéressons aux modèles finis uniquement, nous interprétons une théorie T de $\mathcal{L}$ sur l'ensemble fini $D_n = \{0,\ldots,n-1\}$. Les constantes (entières) sont interprétés par elles mêmes. Nous appelons terme de base ou cellule un terme de la forme $f(e_1,\ldots,e_n)$ où tous les e_i appartiennent à l'ensemble D_n. L'interprétation de toutes les cellules définit une table pour chaque fonction de la théorie T. Un modèle d'ordre n d'une théorie T est une interprétation sur D_n qui satisfait tous les axiomes.

Exemple 1
Soit T_1 la théorie définie par les axiomes suivant :

- $h(x,x) = x$
- $h(h(x,y),x) = y$

T_1 possède plusieurs modèles d'ordre 4 dont celui ci :

h	0	1	2	3
0	0	2	3	1
1	3	1	0	2
2	1	3	2	0
3	2	0	1	3

Cette table est équivalente aux affectations suivantes : $h(0,0) = 0$, $h(0,1) = 2$, $h(0,2) = 3$,...

Nous introduisons maintenant plusieurs concepts nécessaires pour la compréhension de notre algorithme.

Définition 1 *Soit C un sous-ensemble de D_n, et f une fonction. L'image de l'ensemble C par la fonction f, notée $f(C)$, est égale à $\{f(c_i) \mid c_i \in C\}$.*

Définition 2 *Soit g une fonction unaire et I_g une interprétation de g sur $D_n = \{0,\ldots,n-1\}$. Tout sous ensemble minimal pour l'inclusion c de D_n tel que $g(c) = c$ est appelé un cycle. Un élément $i \in D$ apparaît au plus dans un cycle. On définit c_i comme étant le cycle dans lequel i apparaît si celui ci existe et $\emptyset$ sinon. On définit la taille d'un cycle $size(c)$ comme étant égale à $|c| - 1$.*

Définition 3 *Soit g une fonction unaire et I_g une interprétation de g sur $D_n = \{0,\ldots,n-1\}$. Le sous ensemble maximal pour l'inclusion C_{I_g} de D_n tel que $g(C_{I_g}) = C_{I_g}$ est la restriction bijective de g.*

Proposition 1 *Soit g une fonction unaire et I_g une interprétation de g sur $D_n = \{0,\ldots,n-1\}$. Il existe un entier $i \in D_n$ et une permutation σ sur D_n transformant I_g en une interprétation $\sigma(I_g)$ qui vérifie :*

- $C_{\sigma(I_g)} = \{i,\ldots,n-1\}$.
- *pour tout cycle c, les éléments de c sont consécutifs et seul le plus grand élément e de c est tel que $g(e) \leq e$.*
- *deux cycles consécutifs c_1 et c_2 vérifient $size(c_1) \leq size(c_2)$.*

Cette proposition peut aisément être prouvée en construisant itérativement σ par un renommage approprié sur D_n.

Définition 4 *Une interprétation I_g satisfaisant les conditions de la proposition 1 est dite canonique. Soit c un cycle dans I_g. Le plus petit élément de c est appelé $start(c)$ et le plus grand élément $end(c)$.*

En accord avec les précédentes définitions, $size(c) = end(c) - start(c)$ pour tout cycle c d'une interprétation canonique. L'exemple 2 illustre une telle interprétation.

Exemple 2
Soit g une fonction unaire, l'interprétation I_g suivante est canonique :

g	0	1	2	3	4	5	6	7	8	9	10	11
	1	2	2	3	5	4	7	6	9	10	11	8

$C_{I_g} = \{2,\ldots,11\}$ *et I_g contient 5 cycles :*

cycle	size	start	end
0	0	0	0
1	0	1	1
2	1	2	3
3	1	4	5
4	3	6	9

3 Isomorphisme de Modèles

Proposition 2 *Soient T une théorie contenant une fonction unaire g et S un modèle de T canonique sur g. Soit h une fonction de T différente de g et $h(i_1,\ldots,i_k) = v$ une affectation de S. On suppose que $v \in C_{I_g}$ et que $v \notin c_{i_j}$ pour tout i_j. Alors pour tout $w \neq v$ tel que $size(c_w) = size(c_v)$ et tel que w n'appartienne à aucun des c_{i_j}, il existe un isomorphisme transformant S en un modèle S' de T dans lequel $h - i_1,\ldots,i_k) = w$ est vrai.*

Preuve Soient c_v et c_w les cycles, non vides, de v et w. Il y a deux cas possibles :

- $c_v \neq c_w$: Soit $\sigma : D_n \mapsto D_n$, une permutation égale à l'identité partout excepté en c_v et c_w et qui vérifie $g^i(v) = g^i(w)$ et $g^i(w) = g^i(v)$ pour tout $i \in \{0,\ldots,|c_v|-1\}$.

- $c_v \neq c_w$: Soit $\sigma : D_n \mapsto D_n$, une permutation égale a l'identité partout sauf sur c_v et qui vérifie $g^i(v) = g^i(w)$ pour tout $i \in \{0, \ldots, |c_v| - 1\}$.

Par définition, une telle permutation σ vérifie $\sigma(g(i)) = g(\sigma(i))$. On peut aisément appliquer σ aux affectations $h(i_1, \ldots, i_k) = v$ par $\sigma(h(i_1, \ldots, i_k) = v) = h(\sigma(i_1), \ldots, \sigma(i_k)) = \sigma(v)$. Ces conditions associées au fait que σ est bijective et que tous les axiomes universellement quantifiés sont valides sous S assure que chaque axiome terminal $t_1 = t_2$ de la théorie T est valide sous $\sigma(S)$. □

Exemple 3
Supposons que l'on veuille générer des groupes abéliens (voir axiomes à la figure 1) d'ordre 5. Soit I_g l'interprétation canonique suivante :

g	*0*	*1*	*2*	*3*	*4*
	0	*1*	*2*	*4*	*3*

Alors les interprétations $h(1,2) = 3$ et $h(1,2) = 4$ sont isomorphes ; il existe un modèle dans lequel $h(1,2) = 3$ si et seulement si il existe un modèle isomorphe dans lequel $h(1,2) = 4$. Ceci permettra de supprimer la branche associée à $h(1,2) = 4$ de l'arbre de recherche.

4 Génération de modèles

Nous générons les modèles en exploitant la proposition 2 qui permet d'exclure des modèles isomorphes lors de la construction des modèles d'une théorie comportant une fonction bijective g. La construction se fait en deux étapes successives :

4.1 Étape 1

Sélection de la fonction g et génération de ses modèles canoniques conformément à la proposition 1. La construction est très simple :

- $C_{I_g} = \{l, \ldots, n-1\}$ pour un l donné.
- les cycles de I_g ne contiennent que des éléments successifs.
- pour chaque $i \in C_{I_g}$ soit i est la fin d'un cycle, soit $g(i) = i + 1$
- les cycles sont de taille croissante.

Cette procédure ne construit jamais deux interprétations isomorphes de g. Le paramètre l introduit une borne inférieure sous laquelle les symétries ne sont pas détectées. Cela permet de tenir compte des constantes apparaissant dans l'ensemble des axiomes, qui forment un ensemble initial d'éléments non intercheangables. Dans le cas des groupes finis, seule la constante 0 apparaît dans la formulation du problème.

4.2 Étape 2

Pour chaque interprétation I_g générée, l'algorithme énumère tous les modèles possibles de la théorie en utilisant une procédure d'énumération de type CSP, comme cela est décrit dans l'algorithme 1. On dit que l'élément i a été touché par l'algorithme quand :

- l'élément i a été choisi comme valeur d'une cellule.

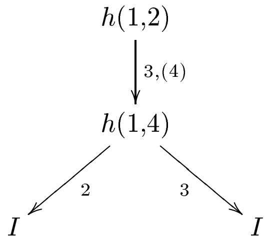
- la cellule $h(i_1, \dots, i_k)$ a été affectée ou est en cours d'affectation et $i = i_j$ pour un $j \in \{1, \dots, k\}$.

On dira également qu'un cycle a été touché si un de ses éléments l'a été. L'algorithme utilise une marque, notée mdn_s, pour chaque cycle de taille s. La valeur de mdn_s représente le max des index de fin des cycles de taille s touchés par l'algorithme. Rappelons que les cycles de taille égale sont consécutifs. Soit $v \in D$, on note $mdn(v)$ la valeur de $mdn_{|c_v|}$. Par la proposition 2 nous savons que quand on choisit une valeur v pour une cellule ce, seules les valeurs plus petites que $mdn(v) + 1$ ont besoin d'être testées. Au départ, tous les mdn sont bornés inférieurement par la plus grande constante utilisée dans les axiomes.

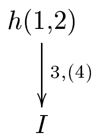
Afin de favoriser l'utilisation de ces propriétés, il est recommandé de choisir les cellules à instancier de manière à ce que les mdn croissent le plus lentement possible. Pour le garantir, il suffit de choisir des cellules ce telles qu'aucun mdn ne change. Lorsque ce n'est plus possible, on choisit une cellule induisant les plus petites modifications possibles des mdn. En pratique il suffit de choisir une cellule non encore évaluée dont les indices sont les plus petits possibles. Cette option généralise l'heuristique LNH de [9]. Nous illustrons le comportement général de cet algorithme par l'exemple suivant :

Exemple 4
Générons tous les modèles des groupes abéliens (figure 1) de taille 5. Seules 3 interprétations canoniques de g sont produites entièrement. Les valeurs qui apparaissent entre parenthèses lors de l'affectation sont celles supprimées par XLNH *du fait de la proposition 2. Les branches n'apparaissant pas sont celles supprimées par la propagation des contraintes (que nous n'avons pas indiquées).* I *désigne l'inconsistance.* M *désigne l'obtention d'un modèle.*

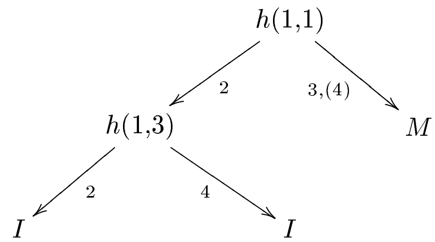
g_1	*0*	*1*	*2*	*3*	*4*
	0	*1*	*2*	*3*	*4*



g_2	*0*	*1*	*2*	*3*	*4*
	0	*1*	*2*	*4*	*3*



g_3	*0*	*1*	*2*	*3*	*4*
	0	*2*	*1*	*4*	*3*



On notera que la première interprétation de g produite étant l'identité, tous les cycles sont donc de même taille (égale à 1) et on se retrouve donc avec l'heuristique LNH *de base alors que la recherche est bien avancée, ce qui permet de supprimer plus d'interprétations isomorphes.*

Algorithme 1 La procédure d'énumération

Fonction Recherche(S : Affectations ; F : Axiomes) : Booléen;

Début
Pour tous les $a \in S$ non propagés **faire** Propager(a,F,S)
Si S contient des affectations incompatibles **Alors Retourner** Faux
Si F est vide **Alors retourner** Vrai
Choisir $b(i_1, \ldots, i_n)$ tel que max(i_j) soit le plus petit possible
Pour chaque i_j **Faire** Mettre à jour mdn(i_j)
Pour chaque $v \in D$ tel que $v \leq mdn(v) + 1$ **Faire**
    Si Recherche($S \cup \{b = v\}$,F) **Alors Retourner** Vrai
Retourner Faux
Fin

---

5 Expérimentations

Nous avons implémenté XLNH sur la base du logiciel SEM dont les sources sont disponibles à l'adresse `http://www.cs.uiowa.edu/~hzhang/sem.html`.

5.1 Description des problèmes

Les problèmes que nous avons expérimentés sont des instances de divers ordres des groupes abéliens, décrits au tableau 1. Les résultats que nous donnons sont en secondes sur un K6II (128Mo, 400Mhz) fonctionnant sous Linux 2.2. Un '+' indique que le problème n'a pas été résolu en moins de deux heures.

5.2 Résultats comparatifs

	SEM + LNH			VESEM			SEM + SYM		
Taille	Modèles	Temps	Noeuds	Modèles	Temps	Noeuds	Modèles	Temps	Noeuds
16	135	3.5	6 109	135	5.5	3 667	75	4	3 630
17	1	5	7 495	1	9	3 962	1	5.5	4 205
18	78	7.5	11 289	78	13	4 565	50	8	6 233

TAB. 1 – *Homogénéité des résultats sur* AG

Le tableau 1 montre sur quelques exemples de groupes abéliens des performances comparables d'approches pourtant différentes. La version standard de SEM [11] est comparée avec VESEM [1] qui utilise une stratégie de type "lookahead" et donne de bons résultats sur certains problèmes (anneaux, groupes non abéliens par exemple). Une comparaison est également faite avec SEM + SYM [2], une évolution de SEM qui recherche dynamiquement des symétries. Le tableau 1 montre clairement que ces trois algorithmes ont des temps d'exécution et des nombres de noeuds similaires.

Le tableau 2 compare la version standard de SEM avec notre méthode. Elle montre que notre approche est 7 fois plus rapide que SEM sur les ordres pairs et jusqu'à 70 fois plus rapide sur les ordres impairs. Il est très intéressant et encore inexpliqué de voir que la génération de modèles de groupes abéliens d'ordre impair est beaucoup plus facile que celle des groupes d'ordre pair. On notera que SEM, qui utilise LNH, ne suggère pas cette propriété. Pour que la comparaison soit réaliste, nous avons rajouté l'axiome de bijectivité de g pour permettre sa prise en compte par SEM (cette propriété est seulement impliquée par les axiomes de base listés en 1). On peut noter que XLNH peut aisément être utilisée dans le cas des fonctions unaires non bijectives, en s'appliquant à la restriction bijective (C_{I_g}).

5.3 Interprétation

L'algorithme proposé est à son avantage dans le cadre de la génération de groupes abéliens car g est non seulement bijective mais en plus elle vérifie $g^2(x) = x$. Tous les cycles sont de taille au plus 1, et nous générons donc au plus n interprétations canoniques de g. Quand une telle interprétation est donnée, l'affectation de la fonction h se fait dans un contexte où interviennent encore beaucoup de symétries. Tous les éléments de cycles de taille 0 (où g est l'identité) sont symétriques entre eux ainsi que tous les éléments de cycles de taille 1. La constante 0, utilisée explicitement dans les axiomes du problèmes, est le seul élément de D_n qui ne soit pas interchangeable au départ.

6 Conclusion

Nous avons proposé une façon de généraliser l'heuristique LNH lorsqu'il existe une fonction unaire bijective dans la théorie. Cette généralisation s'avère très efficace pour réduire le nombre de noeuds parcourus par la procédure d'énumération. Elle permet en outre de générer très peu de modèles isomorphes des groupes abéliens. Un futur travail consistera dans la généralisation de cette approche, afin de résoudre d'autres théories équationnelles et des problèmes plus généraux, comme les théories multi sortes. Une

	SEM + XLNH			SEM + LNH		
Taille	Modèles	Temps	Noeuds	Modèles	Temps	Noeuds
30	34	78	9 654	292	447	160 518
31	1	22	4 450	1	582	166 957
32	529	172	11 904	2 295	956	421 178
33	15	29	5 129	15	1 151	466 883
34	2	242	28 677	20	1 402	481 249
35	13	42	6 333	13	1 700	490 606
36	321	385	31 103	2 142	2 345	872 374
37	1	67	7 524	1	2 848	921 379
38	2	550	43 512	22	3 525	935 527
39	17	88	8 397	17	4 263	946 669
40	282	835	47 245	2 220	5 636	1 393 433
41	1	122	9 913	+	+	+
42	42	1 291	59 952	+	+	+
43	1	158	10 926	+	+	+
44	31	1 839	64 087	+	+	+
45	180	230	12 513	+	+	+
46	2	2 552	67 134	+	+	+
47	1	365	21 433	+	+	+
48	3 345	4 434	75 905	+	+	+
49	8	487	22 976	+	+	+
50	22	6 353	232 718	+	+	+
51	21	632	25 053	+	+	+
52	+	+	+	+	+	+
53	1	833	27 658	+	+	+
54	+	+	+	+	+	+
55	17	1 056	29 875	+	+	+
56	+	+	+	+	+	+
57	23	1 351	32 681	+	+	+
58	+	+	+	+	+	+
59	1	1 659	35 878	+	+	+
60	+	+	+	+	+	+
61	1	2 071	38 005	+	+	+
62	+	+	+	+	+	+
63	278	2 471	40 764	+	+	+

TAB. 2 – *Comparaison de* LNH *et* XLNH

autre étude en cours consiste à combiner cette approche statique avec une méthode de recherche dynamique des symétries.

Références

[1] G. Audemard, B. Benhamou, and L. Henocque. Two techniques to improve finite model search. In *Proceedings of the 11th International Conference on Automated Deduction (CADE-17)*, Pittsburgh, June 2000.

[2] G. Audemard and B. Benhamou. Résultats sur les symétries de théories de la logique du premier ordre. *Research Report*, LIM, 2000.

[3] B. Benhamou and L. Henocque. A hybrid method for finite model search in equational theories. *Fundamenta Informaticae*, 39(1-2):21–38, 1999.

[4] William McCune. A Davis-Putnam program and its application to finite first-order model search: quasigroup existence problems. Technical Memorandum ANL/MCS-TM-194, Argonne National Laboratories, IL/USA, 1994.

[5] Nicolas Peltier. A new method for automated finite model building exploiting failures and symmetries. *Journal of Logic and Computation*, 8:511-543, 1998.

[6] J. Slaney, M. Fujita, and M. Stickel. Automated reasoning and exhaustive search: Quasigroup existence problems. *Computers and Mathematics with Applications*, 29(2):115–132, 1993.

[7] J. Slanley. Finder : Finite domain enumerator. version 3 notes and guides. Technical report, Austrian National University, 1993.

[8] H. Zhang and M. Stickel. Implemanting the Davis-Putnam method. *Journal of Automated Reasonning*, 24:277–296, 2000.

[9] J. Zhang. Constructing finite algebras with Falcon. *Journal of Automated Reasoning*, 17:1–22, 1994.

[10] J. Zhang. Problems on Generation of Finite Models. In *proceedings of cade12, Nancy*, pages 753-757, 1994.

[11] Jian Zhang and Hantao Zhang. SEM: a system for enumerating models. In Chris S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 298–303, 1995.