

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE
SCIENTIFIQUE

UNIVERSITE MOHAMED KHIDER – BISKRA
FACULTE DES SCIENCES ET DES SCIENCES DE L'INGENIEUR
DEPARTEMENT D'INFORMATIQUE

N° D'ORDRE :
SERIE :

VERIFICATION ET DIAGNOSTIC DES SYSTEMES
HARDWARES/SOFTWARES

Thèse présentée pour l'obtention du diplôme de doctorat en informatique

Par

Mr. Mustapha BOURAHLA

Devant le jury composé de :

Dr. BADACHE Nadjib, Prof.	Président	USTHB
Dr. BENMOHAMED Mohamed, Prof.	Rapporteur	Université de Constantine
Dr. BARKAOUI Kamel, Prof.	Examineur	CNAM, Paris
Dr. DJEDI Noureddine, Prof.	Examineur	Université de Biskra
Dr. KOUDIL Mouloud, M.C.	Examineur	INI, Alger
Dr. KIMOUR Mohamed-Tahar, M.C.	Examineur	Université d'Annaba

Thèse soutenue le

خلاصة

التصديق هو مرحلة مهمة في تطوير تصميم الأنظمة المادية أو الحاسوبية. فهي طريقة لإثبات أن نظام موصوف بواسطة برنامج ممثلاً لتصرفه الممكن، ترضي جملة من المميزات ممثلة لتصرفه المرغوب.

عدد من التقنيات التقليدية، مثل التظاهر والاختبار قد أثبتوا محدوديتهم، ولا يمكن أن يكونوا شاملين. في العشرية الأخيرة، يوجد توجه نحو استعمال الطرق الشكلية المعتمدة على الشكليات الرياضية من أجل الفحص الكامل للأنظمة.

نستطيع أن نميز ما بين صنفين كبيرين من الطرق الشكلية. الطرق التي تستعمل البرهان الآلي للنظريات. في هذه الطرق، التصرف الممكن لنظام يوصف بجملة من الفرضيات باستعمال نظرية رياضية مؤسسة بدقة. التصرف المرغوب يوصف بجملة من النظريات. طريقة التفحص تثبت رياضياً هذه النظريات على حسب النظرية الرياضية المؤسسة وجملة الفرضيات.

الصنف الثاني، والذي هو أساس أعمالنا المقدمة في هذه الأطروحة، هو تفحص الشكل الممثل لنظام. التصرف الممكن للنظام يوصف بواسطة شكل، على العموم يكون بشكل آلية. هذا الشكل ينتج من برنامج وصفي مقدم. التصرف المرغوب يوصف بواسطة تعبير منطقي رياضي مؤسس بدقة. ينفذ إجراء آلي من أجل اتخاذ قرار إذا كان الشكل يرضي المميزات (التصرف المرغوب).

توجد مشاكل تعترض تطبيق تقنية تفحص الشكل: يمكن أن يكون وصف نظام غير منته. يمكن أن يكون منته ولكن جدي كبير حيث يتجاوز سعة ذاكرة الحاسب. يمكن لنظام أن يكون معقد أين يصعب (أو يستحيل) إنشاء شكله، مثل الأنظمة ذات الوقت الحقيقي والأنظمة متعددة المؤثرات (الفواعل). الأعمال المقدمة في هذه الأطروحة هي لإعطاء حلول لهذه المشاكل من أجل جعل تقنية تفحص الشكل قابلة للتطبيق لهذه الأنظمة.

من محاسن تقنية تفحص الشكل، استنتاج أمثلة مضادة عندما تكون المميزات غير محققة. هذه الأمثلة المضادة هي آثار لتنفيذات ممكنة تظهر كيف أن الشكل لا يحقق هذه المميزات. المشكل الكبير هو كيف يمكن للمصمم أن يشخص وصف الشكل مستعملاً هذه الأمثلة المضادة. في هذه الأطروحة، صممنا طريقة لإدماج تقنية تفحص الشكل مع تقنية التشخيص بواسطة الشكل لمساعدة المصمم على التشخيص.

Résumé

La validation est une étape importante dans le processus de conception des systèmes hardwares ou softwares. C'est une technique pour montrer qu'un système qui est décrit par un programme représentant son comportement possible, satisfait un ensemble de propriétés représentant le comportement désiré.

Plusieurs techniques traditionnelles comme la simulation et le test, ont montrés leurs limites et elles ne peuvent pas être exhaustives. Dans la dernière décennie, il y a une grande tendance à utiliser des méthodes formelles basées sur des formalismes mathématiques pour la vérification exhaustive des systèmes.

Nous pouvons distinguer deux grandes catégories de méthodes formelles. Les méthodes qui sont basées sur la démonstration automatique des théorèmes. Dans ces méthodes le comportement possible d'un système est décrit par un ensemble d'hypothèses en utilisant une théorie mathématiquement bien fondée. Le comportement désiré est spécifié à l'aide des théorèmes. La technique de vérification est de prouver ces théorèmes selon la théorie de base et l'ensemble des hypothèses.

La deuxième catégorie qui est la base de nos travaux présentés dans cette thèse, est la vérification de modèle. Le comportement possible est décrit par un modèle, en général, sous forme d'automate. Ce modèle est généré à partir d'une spécification d'interface sous forme d'un programme. Le comportement désiré est spécifié par une logique mathématique bien fondée. Une procédure de décision automatique sera exécutée pour décider si le modèle satisfait les propriétés (comportement désiré).

L'applicabilité de la vérification de modèle rencontre des problèmes. Une spécification de système peut générer un modèle (automate) infini (un modèle à une infinité d'états). Un modèle fini peut être tout simplement très large qui dépasse la capacité de la mémoire d'un ordinateur. Un système peut être complexe (ou compliqué) où la construction de son modèle est très difficile (ou impossible) comme les systèmes temps réel et les systèmes multi-agents. Les travaux présentés dans cette thèse sont pour donner des solutions à ces problèmes afin de rendre la technique de vérification de modèle applicable pour ces systèmes.

L'un des avantages de la technique de vérification de modèle est la génération des contre exemples une fois la propriété est insatisfaite. Ces contre exemples sont des traces d'exécutions possibles montrant comment le modèle ne satisfait pas cette propriété. Le grand problème qu'un concepteur rencontre est comment diagnostiquer la spécification de modèle en utilisant ces contre exemples. Nous avons développé une approche intégrant la technique de vérification de modèle avec la technique de diagnostic basé modèle pour aider les concepteurs dans leur diagnostic.

Table des matières

1	Introduction Générale	1
2	Spécification et Vérification formelle	5
2.1	Spécification par la Logique Temporelle	6
2.1.1	La Logique d'Arborescence CTL	6
2.1.2	La Logique CTL*	10
2.1.3	La Logique Temporelle Linéaire	11
2.1.4	La Logique μ -Calcul	12
2.1.5	Invariants et Propriétés de Sûreté	13
2.1.6	Formules de Trajectoire	14
2.2	Spécification avec Modèles à Haut Niveau	14
2.2.1	Mécanismes d'Abstraction	15
2.2.2	Spécification par la Logique	15
2.2.3	Spécification avec Systèmes de Transition	16
2.2.4	Raffinement	18
2.2.5	Automates et Recouvrement des Langages	19
2.3	Conclusion	19
3	Techniques de Vérification et Outils	21
3.1	Vérification de Modèle	21
3.1.1	Vérification de Modèle Enumératif	22
3.1.2	Vérification de Modèle Symbolique	23
3.1.3	Raffinement et Vérification de Modèle	26
3.1.4	Evaluation Symbolique de Trajectoire	27
3.1.5	Outils pour Vérification de Modèle	30
3.2	Approches Basées sur la Théorie des Automates	31
3.2.1	Automates et Recouvrement des langages	32
3.2.2	Recouvrement des Langages et Vérification de Modèle	32
3.3	Méthodes de Dédution	33
3.3.1	Preuve Automatique des Théorèmes	33
3.3.2	Systèmes pour la Démonstration des Théorèmes	34
3.4	Vérification d'un microprocesseur pipeline	35
3.4.1	Preuve de l'Exactitude de Pipeline	37
3.5	Conclusion	38

4	Vérification de Systèmes Infinis	41
4.1	Introduction	41
4.2	Systèmes à État Infini	42
4.3	Interprétation Abstraite	44
4.3.1	Abstractions Basées Fonctions	45
4.3.2	Abstraction par Insertions de Galois	45
4.4	Abstraction de Prédicats	46
4.5	Construction Automatique des Abstractions	47
4.5.1	Génération des Invariants	49
4.5.2	Vérification de Modèle Abstrait	50
4.6	Raffinement Automatique des Abstractions	50
4.7	Conclusion	53
5	Vérification Parallèle et Distribuée	55
5.1	Introduction	55
5.2	Partition des Espaces d'États	56
5.2.1	Abstraction	58
5.2.2	Partition	62
5.2.3	Raffinement	63
5.2.4	Coupure des Transitions Franchissant les Frontières	66
5.3	Vérification Séquentielle de Modèle	67
5.4	Vérification Parallèle et Distribuée de Modèle	69
5.4.1	Exactitude	73
5.5	Conclusion	75
6	Vérification de Systèmes Temps Réel	77
6.1	Introduction	77
6.2	Spécification par les Automates Temporisés	78
6.3	Réduction des Spécifications TCTL	79
6.3.1	Algorithme de Réduction	80
6.3.2	Exactitude de la Réduction	84
6.4	Génération de Système Bi-similaire Fini	85
6.4.1	Bi-simulation Forte	86
6.4.2	Algorithme de Raffinement des Partitions	87
6.4.3	Graphe Quotient	89
6.5	Vérification de Modèle par CTL	90
6.6	Complexité	91
6.7	Conclusion	91
7	Vérification de Systèmes Multi-Agents	93
7.1	Introduction	93
7.2	La logique Temporelle Multi-Agents BDI_{CTL}	94
7.3	Spécification de Systèmes Multi-Agents	96
7.3.1	Sémantiques Formelles	99
7.4	Construction de Modèle des Mondes Possibles	100

7.4.1	Synthèse de Structure Multi-Agents	102
7.5	Vérification de Modèle par BDI_{CTL}	104
7.6	Conclusion	105
8	Diagnostic par Vérification de Modèle	107
8.1	Introduction	107
8.2	Diagnostic Basé Modèle	108
8.2.1	Raisonnement pour le Diagnostic	110
8.3	Calcul des Diagnoses Minimales	114
8.4	Débogage Basé Modèle	117
8.4.1	Un Exemple de Programme VHDL	118
8.4.2	Conversion du Programme	120
8.4.3	Débogage du Programme	124
8.5	Conclusion	125
9	Conclusion Générale	127