THESE HABILITATION

L R I

RAPPORT SCIENTIFIQUE PRESENTE POUR L'OBTENTION D'UNE HABILITATION A DIRIGER DES RECHERCHES

MAGNIEZ F

Unité Mixte de Recherche 8623
CNRS-Université Paris Sud – LRI

05/2007

Rapport de Recherche N° 1471

CNRS – Université de Paris Sud
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bâtiment 490
91405 ORSAY Cedex (France)

N° d'ordre : 1018

Université Paris-Sud
UFR Scientifique d'Orsay

Rapport scientifique présenté pour l'obtention d'une
Habilitation à Diriger des Recherches

par
Frédéric Magniez

Vérification approchée – Calcul quantique

soutenue le 7 mai 2007

Rapporteurs	KANNAN Sampath	Professeur	University of Pennsylvania
	KARPINSKI Marek	Professeur	University of Bonn
	VAZIRANI Umesh	Professeur	University of California, Berkeley
Jury	ASPECT Alain	Directeur de Recherche	CNRS, Institut d'Optique
	COMON Hubert	Professeur	ENS de Cachan
	HABIB Michel	Professeur	Université Paris 7
	KARPINSKI Marek	Professeur	University of Bonn
	SANTHA Miklos,	Directeur de Recherche	CNRS, LRI
	STERN Jacques	Professeur	ENS de Paris

Remerciements

La rédaction de ce mémoire m'a amené à jauger le chemin parcouru depuis l'époque où j'ai soutenu ma thèse et je tiens au terme de ce travail à remercier les personnes qui m'ont accompagné. Si la thèse peut être considérée comme la conclusion du cycle d'études, l'habilitation me paraît constituer le passage symbolique permettant à un chercheur d'être intégré en tant que tel dans la communauté scientifique. J'ai donc choisi de n'inclure dans ce mémoire que les événements postérieurs à ma soutenance de thèse. Il en sera donc de même pour ces remerciements.

Il m'incombe de remercier en tout premier lieu Yannis Manoussakis, collègue et également délégué aux thèses en Informatique, qui a fait preuve d'une grande patience et d'une disponibilité de tous les instants face à mes exigences de date de soutenance. Il a réussi à faire en sorte que l'*impossible administratif* se produise. Cela n'a pas été sans mal pour lui, et je dois aussi remercier au même titre la scolarité de l'Université de Paris-Sud.

J'ai été touché par l'amabilité de tous les membres de mon jury qui me fait l'honneur de leur présence. Chacun d'eux a fait preuve d'une grande disponibilité afin de se libérer pour cette seule date possible du 7 mai ! C'est ainsi que j'ai la grande joie d'accueillir dans ce jury mon directeur de thèse, Miklos Santha, qui m'a formé et avec qui je ne me lasse pas de collaborer. En lui rendant hommage, je voudrais l'assurer de mes efforts pour répondre, sans le décevoir, au soutien sans faille qu'il a manifesté pour me permettre d'obtenir un poste CNRS dans son équipe. Outre Miklos, Jacques Stern est l'unique personne qui était déjà présente dans mon jury de thèse. Sa confiance renouvelée est donc très importante à mes yeux. De plus, une de ses questions lors de ma soutenance de thèse est à l'origine de mon article [**BDH$^+$01**, **BDH$^+$05**] qui a le meilleur indice de citation. Je l'en remercie encore, et je ne pouvais manquer de saisir une nouvelle fois cette occasion. Je remercie tout aussi chaleureusement Alain Aspect, Hubert Comon et Michel Habib d'avoir accepté de rejoindre ce jury. Le large spectre de recherche de chacun des membres du jury couvre les disciplines dans lesquelles je m'inscris, tant pour ce qui concerne les techniques que les portées de mes travaux. La présence de chacun d'eux m'honore à la mesure de la reconnaissance unanime dont ils sont tous l'objet.

J'ai été aussi très heureux d'avoir obtenu l'accord de Sampath Kannan, Marek Karpinski et Umesh Vazirani pour être mes rapporteurs. Je les en remercie encore vivement. Je tiens aussi à mentionner la gentillesse de Marek qui s'est libéré pour faire lui aussi partie de mon jury. Ils sont à eux trois, entre autres, les pionniers de mes domaines de recherche : Sampath est avec Manuel Blum l'inventeur du Self-Testing [BK95], Marek est le précurseur du Property Testing [AKK99], et enfin Umesh a été le premier avec Ethan Bernstein à montrer l'évidence de la supériorité du calcul quantique [BV97].

Parmi les personnes qui m'ont accompagné depuis sept ans, je pense à l'ensemble de mes co-auteurs, tant pour la collaboration fructueuse que pour les liens de sympathie tissés entre nous. Je voudrais citer plus particulièrement ceux avec qui je travaille depuis plusieurs années, et je l'espère pour encore de nombreuses années, à savoir Eldar Fisher, Gábor Ivanyos, Ashwin Nayak et Pranab Sen. Je n'oublie pas parmi eux ceux de mon équipe, Sophie Laplante et Michel de Rougemont. Sophie m'a fait découvrir et aimer plusieurs complexités et Michel n'en finit pas de stimuler notre recherche en vérification approchée. Je dois aussi citer ici Richard Lassaigne, visiteur régulier de l'équipe Algo, pour ses collaborations enrichissantes, mais aussi pour m'avoir fait partager son goût de l'altitude. Je pense aussi à Sylvain Peyronnet avec qui j'ai apprécié de travailler durant sa thèse. Je veux le remercier, ainsi que tous les étudiants que j'ai pu encadrer, d'avoir mené à bien les projets que nous nous étions fixés.

L'environnement de travail de l'équipe Algo est aussi un moteur extraordinaire que je tiens à souligner en remerciant tous ses membres passés et actuels. Je pense tout particulièrement à une de nos figures incontournables, Wenceslas Fernandez de La Vega, dit Lalo. Un problème combinatoire insoluble ? Il en vient à bout, après quelques digressions très *lalosiennes*, grâce à la Méthode probabiliste

qui lui est si chère. Mon seul regret est de n'avoir pu encore écrire un article avec lui, j'espère que cela viendra.

Je n'oublie pas que mon équipe fait parti du LRI, où il fait bon être. Ma gratitude va donc sans retenue à son directeur, Michel Beaudouin-Lafon, à l'équipe administrative et à l'équipe technique, tous d'une gentillesse, disponibilité et réactivité exemplaires.

Je peux maintenant terminer sereinement par celle à qui je pense depuis le début de cette liste, qui me soutient depuis dix ans et s'est déracinée à cause de moi. Elle fait aussi partie de mes relecteurs assidus avec ses parents que je remercie une fois de plus ici.

Table des matières

Table des figures

CHAPITRE 1

Introduction

Mes travaux s'articulent autour de deux grands courants dont les contenus se recoupent. Il s'agit du *property testing*[1] et du *calcul quantique* décrits à la section suivante. Loin d'être isolées, ces thématiques interagissent avec la vérification d'une part et avec l'algorithmique et la complexité d'autre part.

Ces deux domaines trouvent leur raison d'être dans le défi posé par la difficulté liée à la résolution de certains problèmes informatiques. Ils correspondent à deux approches possibles permettant de contourner cette difficulté : simplifier le problème ou accélérer la machine.

Ces deux options ont un point commun : la recherche d'un modèle de calcul rendant efficace le traitement du problème. Deux approches fréquemment considérées consistent à se contenter d'une solution approchée du problème et/ou à recourir à des algorithmes probabilistes, plus rapides que leurs équivalents déterministes. Le property testing et le calcul quantique sont des versions modernes de ces approches.

1. Domaines de recherche

1.1. Property testing $\models_\varepsilon$. La principale motivation pour modifier le modèle de calcul usuel est que la plupart des problèmes de la vie courante sont trop difficiles (par exemple NP-difficile). L'approche traditionnelle consiste à autoriser une approximation aléatoire de la sortie du problème, correspondant à la notion de schéma d'approximation en temps polynomial (PTAS). Cette approche trouve ses limites dans la caractérisation [AS98] de NP par les preuves vérifiables de manière probabiliste (PCP), qui implique des résultats de non-approximabilité [FGL$^+$96, ALM$^+$98]. Par exemple, le nombre chromatique d'un graphe à n sommets ne peut pas être approché à un facteur $n^{1-\varepsilon}$, pour tout $\varepsilon > 0$ [FK98].

L'idée de déplacer l'approximation de la sortie vers l'entrée du problème renvoie à la notion de program checking[1] [BK95, BLR93, RS96] et à PCP [AS98], et a été explicitement étudiée pour les propriétés de graphes dans le contexte du property testing [GGR98]. Le property testing fait partie des algorithmes sous-linéaires : étant donnée une entrée de taille gigantesque, un algorithme sous-linéaire peut décider approximativement une propriété en échantillonant aléatoirement une infime portion de l'entrée. La construction d'algorithmes sous-linéaires est motivée par l'explosion récente de la taille des données des algorithmes utilisés tous les jours dans des applications temps réels, par exemple en bio-informatique pour le décodage du génome, ou pour la recherche de documents dans des bases de données sur Internet. Les algorithmes en temps linéaire, même polynomial, ont longtemps été considérés comme efficaces, mais ce n'est maintenant plus le cas, car les données sont devenues de taille si gigantesque qu'il n'est plus possible de les représenter dans la mémoire vive (RAM) de l'ordinateur, ni même parfois de tout simplement les lire en entier.

Le property testing est plus précisément une notion statistique d'approximation de la vérification d'une propriété sur un objet donné. Introduit dans les années 90 par Blum (prix Turing 1995), Kannan, Luby et Rubinfeld [BK95, BLR93], le property testing s'applique entre autres à l'analyse numérique, aux équations fonctionnelles, à la géométrie, aux statistiques, à la théorie des langages, aux graphes, et au calcul quantique. Le property testing suggère d'approcher un problème de décision (par exemple, la 3-colorabilité) en accord avec une distance donnée sur les objets étudiés (par exemple, les graphes). Deux simplifications sont introduites : une probabiliste et une autre combinatoire. Etant donnée une distance sur les objets considérés (par exemple, le nombre d'arêtes dont diffèrent deux graphes), un ε-testeur pour la propriété accepte tout objet la satisfaisant, et refuse, avec grande probabilité, tout objet à distance *normalisée* (dans l'exemple des graphes, la *proportion* des arêtes dont diffèrent deux

[1] Terme volontairement non traduit afin d'éviter toute ambiguïté avec d'autres termes francophones liés à des notions différentes.

graphes) plus grande que ε de ceux la satisfaisant. Cette non exhaustivité est propre au property testing : la zone d'ambiguïté est appelée *zone grise*. Ces simplifications rendent possible la vérification d'une multitude de propriétés, même NP-complètes, en temps sous-linéaire voire constant une fois $\varepsilon > 0$ fixé. L'idée de normaliser la distance revient pour les graphes à se concentrer sur les graphes denses. Les travaux [AKK99, Fer96] avaient mis en évidence antérieurement le fait que cette restriction rendait facile l'approximation de plusieurs problèmes d'optimisation sur les graphes, qui étaient pourtant difficiles à approcher en général.

Les premiers testeurs [BK95, BLR93] étaient plus précisément des auto-testeurs (self-testers). Les objets sont dans ce cas des fonctions réalisées par des programmes, des circuits ou tout autre dispositif, la distance normalisée entre deux fonctions est la proportion des entrées où elles diffèrent, et une propriété testée peut être, par exemple, « le programme calcule une fonction linéaire ». Les auto-testeurs ont été introduits en même temps que les auto-correcteurs, *i.e.* des procédures transformant un programme correct sur beaucoup d'entrées en un programme probabiliste correct partout. Ainsi si un programme est déclaré suffisamment fiable par un auto-testeur, il peut être transformé en un programme probabiliste correct partout. L'existence d'un couple auto-testeur/correcteur peut paraître surprenante. Il permet d'utiliser avec grande fiabilité des programmes erronés. Un auto-testeur/correcteur doit aussi être plus *simple* que l'objet testé. Par exemple, l'auto-testeur d'un programme doit avoir sa complexité globale négligeable devant celle du programme testé. Donc, en général la complexité du programme testé et corrigé est du même ordre de grandeur que celle du programme erroné. Ce point est fondamental et suggère une implémentation systématique de telles procédures. Les développeurs du logiciel FFTW [FFT], Frigo et Johnson du MIT, ont suivi cette voie en utilisant les travaux d'Ergün [Erg95, EKS00] sur l'auto-test de la transformée de Fourier.

1.2. Calcul quantique $|\psi\rangle$. L'idée du calcul quantique remonte à Richard Feynman (prix Nobel 1965) en 1982. Il avait en effet posé le problème de la simulation d'un système physique quantique par un ordinateur construit selon les règles de la physique classique. Selon lui, le pouvoir de calcul d'une machine utilisant des composants quantiques était susceptible d'être plus important que celui d'une machine classique pour ce problème de simulation. Cette éventualité constitue un nouveau défi à la version quantitative de la thèse de Church-Turing. Cette thèse prédit en effet que tout modèle de calcul physiquement réalisable peut être simulé avec un surcoût polynomial par une machine de Turing probabiliste. Les premiers modèles de calcul quantique, la machine de Turing quantique et les circuits quantiques, ont été définis par Deutsch [Deu85, Deu89]. Yao [Yao93] a montré que ces deux modèles étaient polynomialement équivalents. Bernstein et Vazirani [BV97] ont, entre autre, montré qu'il existait une machine quantique universelle efficace. Tandis que les physiciens cherchent à développer les technologies nécessaires à la construction de telles machines, les informaticiens comparent déjà la puissance du calcul quantique à celle des machines classiques.

Historiquement, le premier résultat d'informatique quantique est apparu pour la cryptographie en 1984. Il s'agit du protocole de distribution de clés secrètes de Bennett et Brassard [BB84], dont Shor et Preskill [SP00] ont démontré ensuite de façon inconditionnelle la sécurité. Ce premier résultat était à la fois surprenant et motivant, puisque rappelons que la sécurité des protocoles classiques repose toujours sur la difficulté de résoudre certains problèmes combinatoires. De plus, ce protocole est implémenté et fiable sur des distances de l'ordre de 100 km.

Mais le résultat qui a le plus marqué et donné son essor à cette discipline est calculatoire. Shor (prix Gödel 1999) a en effet expliqué en 1995 comment factoriser et calculer un logarithme discret en temps polynomial sur un ordinateur quantique [Sho97]. Ces problèmes sont réputés difficiles en informatique classique, c'est pourquoi la sécurité de nombreux protocoles cryptographiques repose empiriquement sur eux : toutes les attaques connues se ramènent essentiellement à factoriser ou à calculer un logarithme discret. Il va donc sans dire qu'en plus de remettre en cause la thèse quantitative de Church-Turing, la construction effective d'un ordinateur quantique aurait aussi de sérieuses répercussions en cryptographie.

L'autre résultat majeur du calcul quantique est celui de Grover [Gro96]. Il a expliqué comment rechercher un élément dans une base de données non structurée (par exemple un tableau non trié) de taille n en seulement $\Theta(\sqrt{n})$ requêtes à la base de données, alors qu'une machine classique (déterministe ou probabiliste) nécessite de l'ordre de $\Theta(n)$ requêtes.

Même si plusieurs résultats viennent appuyer la supériorité du calcul quantique dans des domaines comme la cryptographie, la complexité de communication et les preuves interactives, la puissance du calcul quantique est encore mal cernée. De nombreuses équipes cherchent à trouver d'autres problèmes pour lesquels aucun algorithme efficace n'est connu mais qui pourraient être résolus efficacement sur un ordinateur quantique. Même si un ordinateur quantique n'était jamais réalisé, nous croyons que le calcul quantique apporte aussi un regard neuf à la complexité. C'est ainsi que plusieurs résultats anciens et difficiles de la complexité classique se prouvent élégamment et simplement en utilisant des techniques issues du calcul quantique, par exemple que PP est clos par intersection [Aar05]. Dans certains cas, des problèmes de complexité classique ont même été résolus ou améliorés par des techniques issues du calcul quantique [KW04, SS04, LLS06].

2. Structure du mémoire

Ma recherche en poperty testing est de trois types : construction de nouveaux testeurs, convergence du property testing avec d'autres méthodes de vérification, et enfin apport de la notion du property testing à de nouveaux domaines.

Mes travaux en calcul quantique sont essentiellement concentrés sur la conception de nouveaux algorithmes, ou inversement sur la recherche des limitations du calcul quantique.

J'ai choisi de structurer la présentation de mes travaux en quatre parties dont les publications rattachées sont mentionnées ci-dessous. Lorsque deux références apparaissent, il s'agit d'abord des actes de colloque puis de l'article de revue, sinon il s'agit toujours d'une version actes de colloque, sauf avis contraire.

- Vérification approchée (Chapitre 2) $\models_\varepsilon$
  - Property testing of regular tree languages [**MR04**, **MR06**]
  - Probabilistic abstraction for model checking : An approach based on property testing [**LLM$^+$02**, **LLM$^+$06**]
  - Approximate satisfiability and equivalence [**FMR06**]
- Algorithmique quantique (Chapitre 3) $|\psi\rangle$
  - Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem [**IMS01**, **IMS03**]
  - Hidden translation and orbit coset in quantum computing [**FIM$^+$03**]
  - Quantum algorithms for element distinctness [**BDH$^+$01**, **BDH$^+$05**]
  - Search via quantum walk [**MNRS07**]
  - Quantum algorithms for the triangle problem [**MSS05**, **MSS06**]
  - Quantum complexity of testing group commutativity [**MN05**, **MN06**]
- Bornes inférieures (Chapitre 4) $\models_\varepsilon$ / $|\psi\rangle$
  - Probabilistic abstraction for model checking : An approach based on property testing [**LLM$^+$02**, **LLM$^+$06**] $\models_\varepsilon$
  - Lower bounds for randomized and quantum query complexity using Kolmogorov arguments [**LM04**, **LM06**] $|\psi\rangle$
- Vérification quantique (Chapitre 5) $\models_\varepsilon$ + $|\psi\rangle$
  - Quantum testers for hidden group properties [**FMSS03**] $\models_\varepsilon^{|\psi\rangle}$
  - Self-testing of quantum circuits [**MMMO06**] $|\psi\rangle$ $\models_\varepsilon$

La dernière partie du mémoire (Chapitre 6) sera consacrée à un échantillon des perspectives ouvertes par mes travaux ainsi qu'à l'exposé de nouvelles directions que je compte suivre dans les prochaines années.

Volontairement, mes travaux de thèse [**Mag00a**] sont omis de ce mémoire afin de mettre l'accent sur mes contributions postérieures. Mes travaux sur le self-testing pour le calcul approché [**KMS99**, **KMS03**][**Mag00b**, **Mag05**] font partie d'un article [**KMS00**] de veille scientifique qui a servi de support d'une école d'été à Téhéran. J'ai aussi participé à la rédaction d'un article [**KLM06**] de vulgarisation sur le calcul quantique qu'on pourra consulter lors d'une première approche du domaine. Quant à mon travail sur le self-testing de portes quantiques [**DMMS00**, **DMMS07**], nous aurons l'occasion d'y faire allusion pour présenter notre extension récente pour les circuits quantiques [**MMMO06**].

CHAPITRE 2

Vérification approchée $\models_\varepsilon$

1. Préliminaires

Soit $\mathbf{K}$ une classe de structures finies U, telle que les mots ou les arbres. Une propriété P est un sous-ensemble de $\mathbf{K}$. Une formule F sur $\mathbf{K}$ est définie dans une logique telle que la logique du premier ordre ou la logique monadique du second ordre. La caractérisation logique des propriétés régulières de mots (respectivement d'arbres) par la logique monadique du second ordre sera utilisée. Une structure finie $U \in \mathbf{K}$ *satisfait* P, ou $U \models P$, si $U \in P$. Quand P est définie par une formule F, la notation est étendue à F. A la place des propriétés, les termes classes ou langages seront parfois utilisés, et en particulier, les langages réguliers de mots et d'arbres.

1.1. Distances sur les mots et les arbres. Une *opération élémentaire* sur un mot w est une insertion, un effacement ou une substitution d'une lettre, ou le *déplacement* d'un sous-mot (lettres consécutives) de w à une autre position dans w. La *distance d'édition avec déplacement* $\mathsf{dist}(w, w')$ entre w et w' est le nombre minimal d'opérations élémentaires à effectuer sur w pour obtenir w'.

Cette distance est étendue aux arbres en généralisant la notion d'opérations élémentaires. Une *opération élémentaire* (voir Figure 2.1) sur un arbre ordonné d'arité (nombre maximal de fils par nœud) non bornée (unranked tree) T est une insertion ou un effacement d'un nœud [Tai79], la substitution d'un label, ou le déplacement d'un sous-arbre entier vers un autre nœud de T [**MR04**]. Plus précisément, un *déplacement* (u, v, i) déplace en une étape u (et le sous-arbre correspondant enraciné en u) vers le i-ème successeur de v, déplaçant d'un cran tous les j successeurs de v pour $j \geq i$. Par conséquent, le nouveau père de u est maintenant v. Quand les arbres sont d'arité r (r-ranked tree), les opérations d'effacement et de déplacement sont restreintes de sorte que l'arbre résultant de l'opération est lui aussi d'arité r.

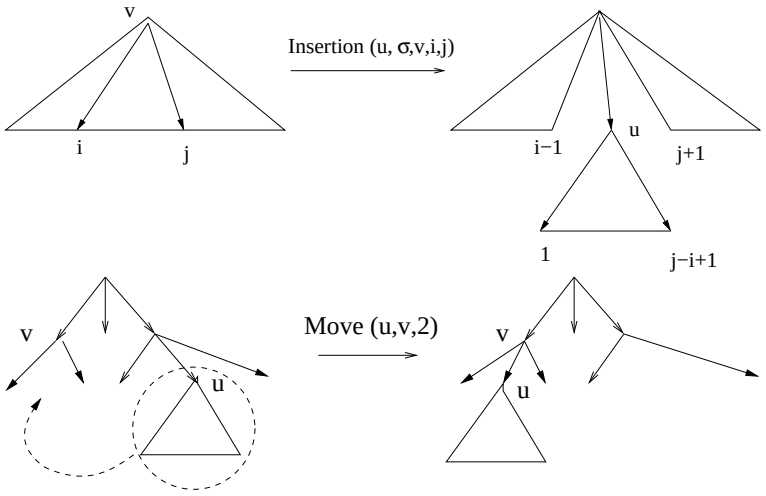

FIG. 2.1. Opérations élémentaires sur les arbres.

1.2. Satisfiabilité et équivalence approchées. La notion de satisfiabilité approchée est définie de manière similaire au property testing [GGR98]. Soit $\mathbf{K}$ une classe de structures finies munie d'une distance dist entre structures. Puisque le property testing est une notion approchée de la vérification pour les instances denses, ou dit autrement pour des distances normalisées, une notion appropriée de proximité est d'abord définie pour chaque distance dist. Deux structures $U, U' \in \mathbf{K}$ sont *ε-proches* si

leur distance est au plus $\varepsilon \times M$, où M est un facteur de normalisation correspondant à la distance maximale $\mathsf{dist}(V, V')$ que peut prendre deux structures $V, V' \in \mathbf{K}$ respectivement de même taille que U, U'. Dans le cas contraire, elles sont alors *ε-éloignées*. Pour les mots et les arbres, la taille maximale des deux structures est choisie pour M, puisque la distance maximale a toujours cet ordre de grandeur. De manière analogue, dans le cas des graphes denses, le carré de la taille maximale des deux structures est choisie pour M.

Définition 2.1. *Soit P une propriété sur $\mathbf{K}$. Une structure $U \in \mathbf{K}$* ε-satisfait *P, ou encore $U \models_\varepsilon P$, si U est ε-proche d'une structure $U' \in \mathbf{K}$ qui satisfait P, i.e. telle que $U' \models P$.*

Quand P est définie par une formule F les définitions et les notations de cette section sont toutes étendues à F. De plus, la notation $U \not\models_\varepsilon P$ sera utilisée pour signifier que U est ε-éloignée de toute structure U' satisfaisant P, *i.e.* telle que $U' \models P$.

Définition 2.2 (Testeur [GGR98]). *Soit $\varepsilon > 0$. Un* ε-testeur *pour une propriété $P \subseteq \mathbf{K}$ est un algorithme probabiliste A tel que, pour toute structure $U \in \mathbf{K}$ fournie en entrée :*
(1) Si $U \models P$, alors A accepte avec probabilité au moins $2/3$;
(2) Si $U \not\models_\varepsilon P$, alors A rejette avec probabilité au moins $2/3$.

Le seuil $2/3$ sur la probabilité d'accepter ou de rejeter est bien sûr arbitraire. Il peut être remplacé par n'importe quelle constante $\gamma > 1/2$, entraînant un facteur multiplicatif en $\log(1/\gamma)$ dans la complexité du testeur. Il suffit en effet d'exécuter $\log(1/\gamma)$ fois un testeur fonctionnant pour le seuil $2/3$, puis de prendre la majorité de ses réponses pour obtenir le seuil γ. De plus, si l'algorithme est garanti de toujours accepter une structure $U \in P$, alors il est appelé un ε-testeur à *erreur d'un coté* (one-sided error). Quand la condition (1) est modifiée comme suit pour un paramètre $0 < \varepsilon_0 < \varepsilon$:
(1') Si $U \models_{\varepsilon_0} P$, alors A accepte avec probabilité au moins $2/3$;
alors le testeur est un *$(\varepsilon_0, \varepsilon)$-testeur (tolérant)* [PRR07]. Les testeurs tolérants sont reliés aux algorithmes d'approximation [PRR07].

Une *requête* à une structure U dépend du modèle pour accéder à cette structure. Pour un mot w, une requête demande la valeur de la i-ème lettre $w[i]$ de w, où i est un entier. Pour T, une requête demande la valeur du label du i-ème nœud de T, où i est un entier, ainsi qu'éventuellement l'indice de son père et de son j-ème successeur, où j est un entier. L'algorithme est aussi supposé pouvoir demander la taille de l'entrée. La *complexité en requêtes* est le nombre de requêtes adressées à la structure. La *complexité en temps* est définie de manière habituelle, où les opérations suivantes sont supposées être effectuées en temps constant : opérations arithmétiques, génération uniforme d'un entier dans un ensemble fini de taille au plus la taille de l'entrée, et requêtes sur l'entrée.

Par définition, la complexité en temps est donc supérieure à la complexité en requêtes. Pour cette raison, la définition suivante ne mentionne que la complexité en temps.

Définition 2.3. *Une propriété $P \subseteq \mathbf{K}$ est* testable*, s'il existe un algorithme probabiliste A tel que, pour tout réel $\varepsilon > 0$ donné en entrée, $A(\varepsilon)$ est un ε-testeur de P, et la complexité en temps de A dépend uniquement de ε.*

Maintenant est présentée la notion d'équivalence approchée de deux propriétés **[FMR06]** dans le cas où ces dernières sont définies par deux formules F_1 et F_2 sur une logique $\mathcal{L}$.

Définition 2.4. *Soit $\varepsilon > 0$. Soient F_1 et F_2 deux formules sur $\boldsymbol{K}$. Alors F_1 est* ε-équivalent *à F_2, ou encore $F_1 \equiv_\varepsilon F_2$, si toute structure $U \in \mathbf{K}$, sauf un nombre fini, qui satisfait $U \models F_1$ satisfait aussi $U \models_\varepsilon F_2$, et réciproquement.*

Définition 2.5 (Testeur d'équivalence). *Soit $\varepsilon > 0$. Un* ε-testeur d'équivalence *(déterministe) d'une logique $\mathcal{L}$ est un algorithme (déterministe) A tel que, étant données en entrée deux formules $F_1, F_2 \in \mathcal{L}$:*
(1) Si $F_1 \equiv F_2$, alors A accepte ;
(2) Si $F_1 \not\equiv_\varepsilon F_2$, alors A rejette.

La version probabiliste de cette définition demanderait aux conditions (1) et (2) ci-dessus d'être vérifiées avec probabilité au moins $2/3$. Nous avons ici cependant choisi une version déterministe, car tous nos testeurs d'équivalence seront déterministes.

2. Nouveaux testeurs

2.1. Langages réguliers d'arbres. Les travaux de [AKNS00] fournissent un testeur de langage régulier de mots en temps indépendant de la taille du mot, pour la distance d'édition classique, *i.e.* sans déplacement.

Nous [**MR04**, **MR06**] avons étendu ces résultats aux langages réguliers d'arbres, *i.e.* ensemble d'arbres reconnus par un automate fini (non déterministe) d'arbre. Un *automate d'arbre* évalue un arbre en partant des feuilles, contenant chacune l'état initial, vers la racine. L'état possible d'un nœud est défini par l'état de ses feuilles suivant l'automate d'arbre. Pour en savoir plus sur les automates d'arbres et leurs applications, on pourra consulter le livre [CDG+97] en préparation. La structure est ici bien plus compliquée que celle des mots. L'existence d'un testeur efficace était une question ouverte [CK02], en partie car le calcul même de la distance d'édition avec déplacement entre deux arbres s'avère être un problème difficile.

Le *problème de la distance d'arbres* prend en entrée deux arbres T_1, T_2 et un entier p, et décide si $\mathsf{dist}(T_1, T_2) \leq p$. Dans le cas de la distance d'édition sans déplacement, le problème est NP-complet pour les arbres non ordonnés, et calculable en temps polynomial pour les arbres ordonnés [Tai79, AG97]. Soit *EDM* le problème de la distance d'arbres pour les arbres ordonnés et la distance avec déplacement.

THÉORÈME 2.1. *EDM est NP-complet pour les arbres d'arité non bornée et pour les arbres d'arité 2.*

Dans un premier temps uniquement les arbres d'arité r sont considérés. Afin de définir le testeur de propriétés régulières d'arbres, la notion de faisabilité d'un sous-arbre pour un automate d'arbre A doit être définie. Soit $G(A)$ le graphe de transitions d'un automate (non déterministe) d'arbre A à m états : $G(A)$ est un graphe dirigé dont les nœuds sont les états de A, et dont les arêtes relient les états connectés par un arbre quelconque (dont la taille peut être choisie bornée par r^m). Sans perte de généralité et pour simplifier, le graphe $G(\mathcal{A})$ est supposé par la suite connexe. Soit $\mathcal{C}(A)$ l'ensemble des composantes connexes de $G(\mathcal{A})$.

De manière générale, un *sous-arbre* est un arbre ayant des nœuds marqués indiquant à quels endroits il a été 'coupé'. *Compléter* un sous-arbre signifie l'inscrire dans un arbre en fonction de ses marques.

DÉFINITION 2.6. *Soit $\Pi \subseteq \mathcal{C}(A)$. Un sous-arbre t est* Π-faisable *s'il est possible de le compléter de sorte que l'automate A l'accepte en ne parcourant uniquement que des états des composantes de Π.*

Regular ranked tree language tester (A, ε, T)

(1) Soit m le nombre d'états de A
(2) Calculer $N = \lceil (m+1) \times 64 r^{4m+2}/\varepsilon^2 \rceil$
(3) Pour $i = 1, \ldots, 16 r^{2m+1}/\varepsilon$
  - Choisir aléatoirement N nœuds v_j^i, pour $j = 1, \ldots, N$
  - Lire le sous-arbre t_j^i de T depuis v_j^i à profondeur i, pour $j = 1, \ldots, N$
(4) Pour tout $\Pi \subseteq \mathcal{C}(A)$
  - Si chacun des sous-arbres t_j^i est Π-faisable alors accepter (et stopper)
(5) Rejeter

THÉORÈME 2.2. *Pour tout réel $\varepsilon > 0$, tout automate (non déterministe) d'arbre d'arité r à m états, et tout arbre d'arité r, l'algorithme* ***Regular ranked tree language tester*** (A, ε, T) *est un ε-testeur du langage reconnu par A. De plus, sa complexité en temps est en $r^{O(r^{2m+1}/\varepsilon)}$.*

COROLLAIRE 2.1. *Les propriétés régulières d'arbre d'arité bornée sont testables.*

Ce testeur s'applique aussi aux mots en simplifiant et améliorant le testeur antérieur [AKNS00]. Cependant cette amélioration est à nuancer par le fait que ce testeur était construit pour la distance d'édition sans déplacement.

THÉORÈME 2.3. *Il existe un algorithme probabiliste qui, pour tout réel $\varepsilon > 0$ et tout automate A (non déterministe) à m états pris en entrée, est un ε-testeur du langage reconnu par A. De plus, sa complexité en requêtes est en $O(m^3 \log^2(m^2/\varepsilon)/\varepsilon)$, et sa complexité en temps en $O(2^m m^5 \log^2(m^2/\varepsilon)/\varepsilon)$.*

Corollaire 2.2. *Les propriétés régulières de mots sont testables.*

La dernière étape consiste à étendre le testeur pour les arbres d'arité bornée aux arbres d'arité non bornée. La méthode consiste à encoder de manière naturelle un arbre T d'arité non bornée en un arbre $e(T)$ d'arité 2 selon la Figure 2.2.

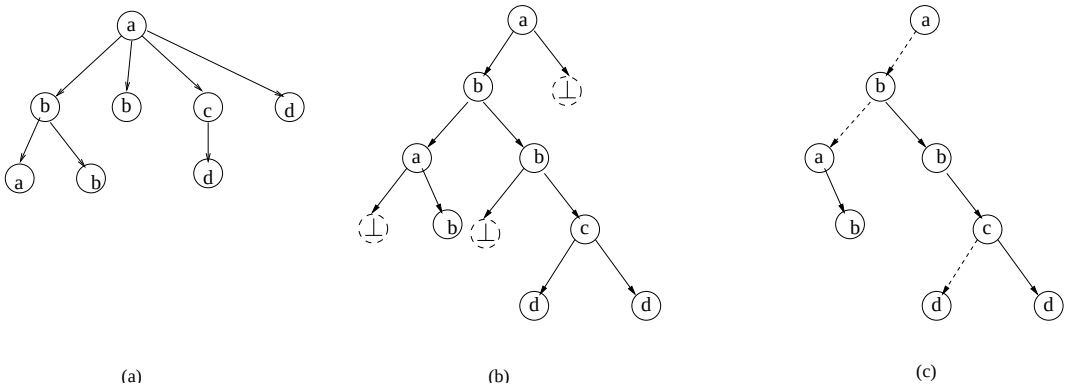

Fig. 2.2. Encodage d'un arbre T d'arité non bornée (a), en un arbre binaire avec des feuilles $\perp$ (b), et en un arbre d'arité 2 dont les nœuds ont un fils droit et/ou un fils gauche (c).

Cet encodage préservant les distances respectives entre arbres par le lemme suivant.

Lemme 2.1. *Pour tout arbres T and T' d'arité non bornée,*

$$\mathsf{dist}(e(T), e(T'))/3 \leq \mathsf{dist}(T, T') \leq 3 \times \mathsf{dist}(e(T), e(T')).$$

De plus, il est possible d'échantillonner un sous-arbre à profondeur k dans $e(T)$, où k est un entier, directement depuis T. Par conséquent notre testeur pour les arbres d'arité bornée s'étend directement aux arbres d'arité non bornée.

Corollaire 2.3. *Les propriétés régulières d'arbres d'arité non bornée sont testables.*

La conclusion de ce travail est donc que la distance d'édition avec déplacement est naturelle pour le property testing. Elle s'adapte naturellement des mots aux arbres, et permet de définir dans les deux cas des testeurs efficaces. Ce résultat a d'importantes applications potentielles : on peut en effet voir le déroulement d'un protocole à plusieurs joueurs comme un arbre, ou un jeu sous forme extensive. Si on peut tester une propriété régulière de cet arbre, on teste indirectement une propriété du protocole. De plus, une application pratique est aussi possible. En effet, toute structure XML peut se voir comme un arbre devant satisfaire certaines contraintes dictées par un automate. Actuellement, les vérifications de syntaxe sont à la fois lourdes et incapables de corriger une structure endommagée. Nous espérons que l'existence d'un testeur mais aussi d'un correcteur est une voie vers une solution à ces deux problèmes.

2.2. Estimateurs de distance. Calculer la distance d'édition avec déplacement est un problème NP-difficile, alors que calculer la distance d'édition simple est globalement quadratique. Dans le premier cas, il existe [CM02] un estimateur de la distance entre deux mots en temps sous-quadratique avec un facteur d'approximation logarithmique. Dans le deuxième cas, il existe [BEK$^+$03] un estimateur au sens du property testing qui accepte les paires de mots qui sont à distance sous-linéaire en leur taille, et rejette ceux qui sont à distance linéaire. La complexité de cet estimateur est aussi sous-quadratique.

Nous [**FMR06**] avons amélioré l'estimateur de la distance avec déplacement au sens de ce qui a été fait pour la distance d'édition simple. Nous avons obtenu un algorithme de complexité constante, *i.e.* ne dépendant que des seuils d'approximations souhaités et non de la taille des mots considérés.

Uniform Tester(w, w', ε)

(1) Calculer $N = \Theta(\frac{(\ln|\Sigma|)|\Sigma|^{2/\varepsilon}}{\varepsilon^3})$ et $k = \frac{1}{\varepsilon}$
(2) Choisir aléatoirement N indices dans $\{1, \ldots, n-k+1\}$
(3) Lire les sous-mots de longueur k de w et w' aux positions ci-dessus
(4) Calculer les statistiques d'occurrences de ces sous-mots $\widehat{\mathsf{u\text{-}stat}}_N(w)$ et $\widehat{\mathsf{u\text{-}stat}}_N(w')$
(5) Accepter si $\left|\widehat{\mathsf{u\text{-}stat}}_N(w) - \widehat{\mathsf{u\text{-}stat}}_N(w')\right| \leq 6.25\varepsilon$
(6) Rejeter sinon

THÉORÈME 2.4. *Pour tout $\varepsilon > 0$, et tous mots w, w' de même taille,* **Uniform Tester**(w, w', ε)
(1) accepte si $w = w'$ avec probabilité 1 ;
(2) accepte si w et w' sont ε^2-proches avec probabilité au moins $2/3$;
(3) rejette si w et w' sont 5ε-éloignés avec probabilité au moins $2/3$.
De plus la complexité en temps est en $O(\frac{(\ln|\Sigma|)|\Sigma|^{2/\varepsilon}}{\varepsilon^4})$.

Cet estimateur a été étendu aux arbres, résolvant ainsi le problème du test d'isomorphisme d'arbres en temps constant. Ce résultat est d'autant plus surprenant que le même problème n'est pas testable pour les graphes denses [FM06].

THÉORÈME 2.5. *Pour tout $\varepsilon > 0$, le problème d'isomorphisme d'arbres est $(\varepsilon^4, O(\varepsilon))$-testable avec une complexité en temps en $|\Sigma|^{O(1/\varepsilon^5)}$.*

Pour cela, nous avons défini un plongement statistique des mots/arbres dans ℓ_1, qui généralise la transformation originale de Parikh [Par66], obtenu par échantillonage aléatoire d'une petite portion des mots/arbres.

Les *satistiques uniformes* $\mathsf{u\text{-}stat}(w)$ correspondent aux statistiques des occurrences des sous-mots de taille k de w pris à des positions aléatoires dans w. Cette notion est très proche de celle du travail antérieur de [Bro97], où les sous-mots de longueur k sont appelés "shingles".

Par exemple, pour les mots binaires, si $k = 2$ (et donc $\varepsilon = 0.5$), il existe quatre blocs possibles de longueur 2, classés par la suite selon l'ordre lexicographique. Ainsi pour le mot binaire $w = 000111$, $\mathsf{u\text{-}stat}(w) = (2/5, 1/5, 0, 2/5)$, car il y a 2 blocs 00, 1 bloc 01, pas de bloc 10 et 2 blocs 11 parmi les 5 sous-mots de w.

Dans ce qui suit $|\cdot|$ représente la norme ℓ_1.

LEMME 2.2. *Soit $n = \Omega(\frac{1}{\varepsilon})$. Si $\mathsf{dist}(w, w') \leq \varepsilon^2 n$ alors $|\mathsf{u\text{-}stat}(w) - \mathsf{u\text{-}stat}(w')| \leq 6.1\varepsilon$.*

LEMME 2.3. *Soit $n = \Omega(\frac{(\ln|\Sigma|)|\Sigma|^{2/\varepsilon}}{\varepsilon^5})$. Si $\mathsf{dist}(w, w') \geq 5\varepsilon n$ alors $|\mathsf{u\text{-}stat}_k(w) - \mathsf{u\text{-}stat}_k(w')| \geq 6.5\varepsilon$.*

La prolongation de ce travail consiste à montrer que des langages ou propriétés deviennent testables pour la distance d'édition avec déplacement alors qu'ils ne l'étaient pas pour la distance d'édition simple. C'est un des buts de notre travail **[FMR06]** que nous détaillons dans la section suivante.

3. Convergence de méthodes de vérification

Si la nécessité de tester un programme, un protocole, ou une machine va de soi, les méthodes pour y parvenir sont beaucoup moins évidentes. Chaque situation motive une méthodologie bien précise. Aussi, il serait déraisonnable de louer une école plutôt qu'une autre. Il est aussi faux de penser que les communautés qui en découlent ne peuvent interagir. Au contraire, comme le montrent plusieurs travaux dont les nôtres, seules de telles interactions peuvent parfois permettre de résoudre des problématiques jusque là restées ouvertes.

Notre travail, commencé il y a cinq ans, consiste à illustrer le potentiel des outils d'approximation probabilistes venant du property testing pour des problèmes de vérification de type model checking (voir [BBF+01] pour une présentation du model checking).

Une des applications de la logique à l'information et la communication concerne la vérification formelle de programmes et de protocoles informatiques, considérés comme des systèmes de transitions pour lesquels certaines propriétés définies dans une logique sont à vérifier. Le model checking est une

méthode automatique permettant de décider si un système de transitions satisfait une spécification exprimée par une formule de la logique temporelle. Cependant, dans de nombreuses situations concrètes, le principal problème est l'explosion combinatoire du système de transitions par rapport à la représentation originale (le programme ou le protocole).

Dans un premier temps, nous avons défini et validé le concept d'abstraction probabiliste de systèmes déterministes, permettant d'utiliser des méthodes du model checking là où elles échouaient. Dans un deuxième temps, nous avons étendu la notion de satisfiabilité approchée inhérente au property testing à celle d'équivalence approchée de structures finies, ayant pour application potentielle la vérification approchée de programmes.

3.1. Abstraction probabiliste. Nous [**LLM+02**, **LLM+06**] avons travaillé sur une utilisation du property testing pour la simplification de la vérification en model checking. Lors de l'utilisation du model checking pour vérifier un programme sur toutes ses entrées, le principal obstacle rencontré est la grande taille de la représentation (par exemple en OBDD) du système de transitions associé au programme, le rendant intraitable.

Nous avons réuni les points forts de ces deux approches en introduisant la notion d'abstraction probabiliste et en étendant le concept du model checking pour pouvoir utiliser ces abstractions. Nos abstractions sont construites sur la notion d'approximation du property testing. Cette approximation est à elle seule insuffisante pour simplifier la tâche du model checking, et doit être accompagnée d'une abstraction probabiliste. En effet, nous avons montré que les systèmes de transitions des propriétés approchées pouvaient garder une représentation en OBDD de taille exponentielle, alors qu'elle devenait constante après abstraction probabiliste. Cette preuve utilise des méthodes venant de la théorie de la complexité de la communication qui pourraient être réutilisées à d'autres fins (voir Chapitre 4).

3.1.1. *ε-réductibilité.* Nous considérons maintenant uniquement les graphes simples (sans arêtes multiples) et non orientés. Pour un graphe G, l'ensemble de ses sommets est noté V_G, celui de ses arêtes E_G, et n le cardinal $|V_G|$ de V_G. Si le contexte est sans ambiguïté, l'indice G et V_G et E_G pourra être supprimé. La distance $\mathsf{dist}(G, G')$ entre deux graphes G and G' ayant le même ensemble de n sommets V est le nombre minimal d'arêtes qu'il faut modifier sur G pour obtenir G'. A cause de la normalisation de la distance, G et G' seront dits ε-proches si $\mathsf{dist}(G, G') \leq \varepsilon n^2$, et ε-éloignés sinon.

La notion d'ε-réduction, implicitement présente dans la plupart des testeur de propriétés de graphe, est centrale à la construction de nos abstractions probabilistes. Intuitivement cette notion signifie qu'il est possible de réduire l'ε-satisfiabilité d'une propriété sur tous le graphe en vérifiant une autre propriété (parfois la même) mais sur un petit sous-graphe aléatoire.

Pour tout graphe G et tout entier k, soit Π la famille des sous-ensembles $\pi \subseteq V_G$ de taille k. Donc Π dépend à la fois de k et V_G. Soit G_π le sous-graphe induit par G sur un sous-ensemble de sommets $\pi \subseteq V_G$.

Définition 2.7. *Soient un réel $\varepsilon > 0$, un entier $k \geq 1$, et deux propriétés de graphe ϕ, ψ. Alors ϕ est (ε, k)*-réductible à *ψ si et seulement si pour tout graphe G,*

$$\begin{aligned} G \models \phi \quad &\Longrightarrow \quad \forall \pi \in \Pi,\ G_\pi \models \psi, \\ G \not\models_\varepsilon \phi \quad &\Longrightarrow \quad \Pr_{\pi \in \Pi}[G_\pi \models \psi] \leq 1/3. \end{aligned}$$

La propriété ϕ est dite *ε-réductible à ψ* s'il existe une constante k telle que ϕ est (ε, k)-réductible à ψ. La testabilité de la c-colorabilité and et bipartition [GGR98, AK02] en termes de ε-réductibilité.

Théorème 2.6 ([AK02]). *Pour tout $c \geq 3$, $\varepsilon > 0$,*

(1) *c-colorabilité est $(\varepsilon, O((c \ln c)/\varepsilon^2))$-réductible à c-colorabilité ;*

(2) *bipartition est $(\varepsilon, O((\ln^4(\frac{1}{\varepsilon}) \ln \ln(\frac{1}{\varepsilon}))/\varepsilon))$-réductible à bipartition.*

Alon, Fischer, Krivelevich et Szegedy [AFKS00] ont montré que toute propriété de graphe du premier ordre de la fomr $\exists\forall$ admettait un ε-testeur. Ce résultat peut aussi s'écrire avec la notion d'ε-réductibilité comme suit.

Théorème 2.7 ([AFKS00]). *Toute propriété de graphe du premier ordre de la forme $\exists\forall$ est ε-réductible à une propriété de graphe.*

3.1.2. *Contexte.* Dans l'exemple suivant, est envisagé le cas d'un programme supposé calculer une fonction booléenne sur des graphes de taille bornée. La vérification porte sur la relation entre l'acceptation du programme et une propriété de graphe ϕ, décrite par la spécification suivante :

Le programme P accepte uniquement les graphes satisfaisant ϕ.

Afin de formaliser une telle spécification, le vocabulaire du model checking est utilisé. A un programme P est associé un système de transitions $M = \langle S, I, R\rangle$, où un état $s \in S$ est une suite finie de variables de P, l'ensemble $I \subseteq S$ représente les états initiaux de P, et R est l'ensemble des transitions possibles correspondant aux étapes élémentaires de P.

Si $\mathtt{G}$ est une variable d'entrée de P telle que $\mathtt{G}$ est interprété comme un graphe G (selon un encodage fixé), alors on écrira $\mathtt{G} = G$. Pour tout graphe G, le sous-ensemble des états initiaux correspondant à la donnée de G en entrée est défini par $I_G = \{s \in I : \mathtt{G} = G\}$. Formellement, la spécification de notre exemple se réécrit alors comme suit :

$$\forall G\Big(\big(\forall s \in I_G \quad M, s \models \exists\, \big((\neg \mathtt{ack})\, \mathbf{U} (\mathtt{ack} \wedge \mathtt{ret})\big)\big) \quad \Longrightarrow \quad G \models \phi\Big).$$

Dans la partie droite de cette implication, noter que ϕ est interprété dans la structure du graphe G qui n'inclut pas le système de transitions. Ceci est rendu nécessaire par le fait que les algorithmes standards de model checking ne sont utilisables que pour les programmes dont les entrées sont des structures du premier ordre. Quand le contexte ne sera pas ambigu, l'expression simplifiée $M, G \models \Theta$ sera utilisée à la place de $\quad \forall s \in I_G, \quad M, s \models \Theta$.

Plus généralement, notre contribution s'applique aux formules du type :

$$\forall G \quad (M, G \models \Theta \quad \Longrightarrow \quad G \models \phi), \tag{2.1}$$

où l'entrée inclu le graphe G ainsi qu'éventuellement d'autres données auxiliaires, Θ est une formule de CTL*, et ϕ est une propriété de graphe.

Dorénavant, un graphe G sera toujours supposé être une des variables du programme. Puisque G sera de taille bornée et ϕ une formule exprimant une propriété de graphe, il est possible de décider si $G \models \phi$ en utilisant des techniques à bases d'OBDD. Soit $\mathrm{sat}(\phi, G)$ un tel OBDD qui s'évalue à vrai si et seulement si $G \models \phi$. Alors vérifier la spécification (2.1) peut être effectué en vérifiant la validité de la formule $\Big(\big(\neg \mathcal{I}_G \vee \mathrm{check}(\mathcal{R}, \Theta)\big) \implies \mathrm{sat}(\phi, G)\Big)$, où $\mathcal{I}_G = \mathcal{I}(G/\mathtt{G})$ (*i.e.*, toutes les occurances de la variables $\mathtt{G}$ sont substituées par G).

Pour les propriétés de graphe que nous considérons, telle que la bipartition, les OBDD pour $G \models \phi$ ont une taille exponentielle. Comme nous le montrons au Chapitre 4, la simplification apportée par le property testing n'est pas suffisante pour réduire la taille de l'OBDD pour bipartition, en partie car elle n'apporte une vérification que pour une entrée (ici un graphe) donnée. Cependant, l'utilisation de l'ε-réductibilité permet de construire des abstractions probabilistes aboutissant à des OBDD plus petits, et même de taille constante. L'utilisation de ces OBDD permet de garantir qu'un programme satisfait approximativement la spécification (2.1) sur toutes ses entrées.

3.1.3. *Abstraction probabiliste.* Voici le model checker incorporant l'abstraction probabiliste correspondant à la vérification de la spécification (2.1). Plus précisément, pour tout sous-ensemble de sommets $\pi \in \Pi$, une abstraction est construite. Dans le nouveau système abstrait $\widehat{M}^\pi$ de M chaque variable v et constante d est notée $\widehat{v}^\pi$ and $\widehat{d}^\pi$. L'unique contrainte est que dans $\widehat{M}^\pi$, le graphe G soit exactement abstrait en G_π, *i.e.* $\widehat{G}^\pi = G_\pi$.

Graph Test$\big((\Pi, \mathcal{M}), \Theta, \psi\big)$

(1) Choisir aléatoirement un sous ensemble de sommets $\pi \in \Pi$.
(2) Accepter si et seulement si
$\forall \widehat{G}^\pi \quad (\widehat{M}^\pi, \widehat{G}^\pi \models \mathcal{D}(\Theta) \quad \Longrightarrow \quad \widehat{G}^\pi \models \psi)$.

Le théorème suivant établit que si Θ est une formule du fragment $\exists$CTL* et si ϕ est ε-réductible à ψ, alors l'abstraction probabiliste ci-dessus permet de garantir sa validité.

THÉORÈME 2.8. *Soit Θ une formule du fragment $\exists$CTL*. Soient un réel $\varepsilon > 0$, un entier $k \geq 1$, et ϕ une formule (ε, k)-réductible à ψ. Si $\big(\exists G \quad (M, G \models \Theta$ et $G \not\models_\varepsilon \phi)\big)$ alors* ***Graph Test****$\big((\Pi, \mathcal{M}), \Theta, \psi\big)$ rejette avec probabilité au moins* $2/3$.

La réciproque peut aussi être établie avec des conditions plus fortes mais néanmoins réalistes qui ont été testées sur un exemple concret correspondant à la bipartition dans [**LLM$^+$02**, **LLM$^+$06**].

3.2. Equivalence approchée entre structures finies. Nous montrons dans cette partie, que la notion d'équivalence approchée, construite sur celle de la satisfiabilité approchée du property testing, peut être décidée efficacement dans plusieurs situations importantes où le cas exact est difficile voire indécidable. Le contexte est celui des mots et des arbres munis de la distance d'édition avec déplacement.

Les statistiques des sous-mots de mots reconnus par un langage vont être utilisées pour caractériser ce langage. Plutôt que de considérer les statistiques uniformes d'un mot, des statistiques plus simples à manipuler vont d'abord être étudiées. Toutefois, tout ce qui suit peut être étendu aux statistiques uniformes, permettant d'obtenir des testeurs tolérants.

Dorénavant, tout mot w est implicitement décomposé en sous-mots consécutifs de k lettres, *i.e.* $w = w[1]_b w[2]_b \dots w[\varepsilon n]_b$ où $w[i]_b \in \Sigma^k$ est le *i-ème bloc* de w. Le paramètre k est un paramètre qui est relié au paramètre d'approximation ε par $k = 1/\varepsilon$. Les *statistiques de blocs* $\mathsf{b\text{-}stat}(w)$ sont les statistiques d'occurrences de chaque bloc de w.

La caractérisation d'un mot w par ses statistiques de blocs $\mathsf{b\text{-}stat}(w) \in \mathbb{R}^{|\Sigma|^{1/\varepsilon}}$ est approximativement bijective si la taille des mots est fixée. Ceci implique qu'étant donnée une puissance de pré-calcul illimité, tout langage peut être testé avec une complexité en requêtes constante. Effectivement, il suffit de pré-calculer les statistiques de blocs de chaque mot de taille n du langage, où n est la taille du mot testé. Cependant, il n'est pas difficile de construire à l'aide de puissances appropriées un langage dont le testeur requiert une complexité en temps arbitrairement grande.

Notre attention va donc être portée sur un pré-calcul efficace ne dépendant pas de la taille des mots testés. Pour cela, commençons par remarquer que les statistiques de bloc ne permettent pas de différencier des mots de tailles différentes puisque $\mathsf{b\text{-}stat}(w_0) = \mathsf{b\text{-}stat}(w_0^t)$ pour tout entier $t \geq 1$, si w_0 est un mot dont la taille est multiple de k. En conséquence, les statistiques de bloc $\mathsf{b\text{-}stat}(w)$ des mots $w \in L$ ne sont pas une bonne caractérisation d'un langage L en général. Par exemple, le mot $w_0^{3\times 2^{s-1}}$ est $(1 - 1/k^{2^{s-1}})$-éloigné du langage $\{w_0^{2^t} : t \geq 1\}$, pour tout entier $s \geq 1$, alors qu'il a les mêmes statistiques de bloc que les mots de ce langage.

Pour construire un testeur qui fonctionne pour toute longueur de mot avec un seul pré-calcul, il faut considérer les statistiques des boucles du langage, au sens d'un lemme de pompage adapté. Ceci prend tout son sens pour des langages composés de boucles. Les langages réguliers ont bien entendu cette propriété, ainsi que les langages algébriques lorsque la permutation des blocs est autorisée. Cette dernière condition prend tout son sens dans notre contexte, puisqu'un mot a au plus n/k blocs, il est ε-proche de toutes les permutations possibles de ses blocs si $\varepsilon = 1/k$.

Fixons maintenant un automate (non déterministe) A à m états et reconnaissant un langage L. Le plongement géométrique de L est alors défini par la réunion des enveloppes convexes de ses boucles *compatibles*, *i.e.* des boucles pouvant apparaître simultanément dans un mot du langage.

Définition 2.8. *Soit $\mathcal{H}$ l'union des* $\mathsf{Convex\text{-}Hull}(\mathsf{b\text{-}stat}(v_1), \dots, \mathsf{b\text{-}stat}(v_t))$ *où $v_1, \dots, v_t$ parcourt les boucles compatibles de A dont la longueur est un multiple de k, et $t \geq 0$.*

Tous les éléments sont alors en place pour énoncer le pouvoir de caractérisation de A par $\mathcal{H}$.

Théorème 2.9. *Soient $w \in \Sigma^n$ et $X \in \mathcal{H}$ tels que $|\mathsf{b\text{-}stat}(w) - X| \leq \delta$. Alors*

$$\mathsf{dist}(w, L) \leq \Big(\tfrac{\delta}{2} + \big(1 + O(\tfrac{m|\Sigma|^{1/\varepsilon}}{\varepsilon^2 n})\big)\varepsilon\Big)n.$$

Le testeur du langage L reconnu par A découle alors du fait qu'une approximation suffisante de $\mathcal{H}$ peut être calculée en temps polynomial à k fixé , *i.e.* $\varepsilon > 0$ fixé. Ce théorème est à comparer au Théorème 2.3 précédemment décrit. Ici la dépendance en la taille de l'automate sert uniquement en pré-calcul pour la construction du testeur.

Théorème 2.10. *Pour tout réel $\varepsilon > 0$ et tout langage régulier L sur un alphabet fini Σ, il existe un ε-testeur de L dont la complexité en requêtes est en $O(\frac{(\ln|\Sigma|)|\Sigma|^{2/\varepsilon}}{\varepsilon^4})$ et la complexité en temps en $2^{|\Sigma|^{O(1/\varepsilon)}}$.*
De plus, étant donné un automate (non déterministe) à m états reconnaissant L, le testeur peut être construit en temps $m^{|\Sigma|^{O(1/\varepsilon)}}$.

Le plongement statistique dont nous avons parlé plus haut permet de tester l'ε-équivalence entre deux propriétés regulières sur les mots, définies par des formules monadiques du second ordre. La complexité de notre testeur est polynomiale en la taille de l'automate (ou de l'expression régulière), pour un ε fixé, alors que la version exacte est PSPACE-complète.

THÉORÈME 2.11. *Il existe un algorithme déterministe T tel que, pour tout $\varepsilon > 0$ pris en entrée, $T(\varepsilon)$ est un ε-testeur d'équivalence pour les automates (non déterministes) définis sur un alphabet fini Σ. De plus, la complexité en temps de T est en $m^{|\Sigma|^{O(1/\varepsilon)}}$, où m est le nombre d'états des automates donnés en entrée du testeur.*

Nos testeurs s'étendent aux langages réguliers infinis et aux grammaires algébriques.

THÉORÈME 2.12. *Il existe un algorithme déterministe T tel que, pour tout $\varepsilon > 0$ pris en entrée, $T(\varepsilon)$ est un ε-testeur d'équivalence pour les automates (non déterministes) de Büchi définis sur un alphabet fini Σ. De plus, la complexité en temps de T est en $m^{|\Sigma|^{O(1/\varepsilon)}}$, où m est le nombre d'états des automates donnés en entrée du testeur.*

THÉORÈME 2.13. *Les propriétés algébriques de mots sont testables.*

THÉORÈME 2.14. *Il existe un algorithme déterministe T tel que, pour tout $\varepsilon > 0$ pris en entrée, $T(\varepsilon)$ est un ε-testeur d'équivalence pour les grammaires algébriques définies sur un alphabet fini Σ. De plus, la complexité en temps de T est exponentielle en $m^{|\Sigma|^{O(1/\varepsilon)}}$, où m est le la taille des grammaires données en entrée du testeur.*

La preuve de ces deux résultats utilise le théorème original de Parikh [Par66], qui fournit une formule définissant un ensemble de contraintes semi-linéaires sur les occurrences possibles des lettres des mots d'un langage algébrique donné. A partir de là, il est possible de définir un automate fini qui reconnait le même langage à permutation des blocs près. L'explosion exponentielle vient de cette étape.

La première extension a une application directe à la logique temporelle linéaire (LTL). Une construction classique associe un automate de Büchi à une formule LTL, dont la taille est exponentielle en celle de la formule. Quand le model checking exact est impossible, on peut donc utiliser notre testeur d'équivalence pour un paramètre d'approximation ε fixé. Pour les grammaires algébriques, le testeur d'équivalence a une complexité exponentielle, alors que la version exacte est indécidable. Enfin, les grammaires algébriques sont testables, alors qu'elles ne le sont pas dans le cas général pour la distance d'édition sans déplacement.

Pour finir, il est possible d'étendre ces résultats aux arbres, cependant le testeur d'équivalence obtenu n'est pas meilleur que celui correspondant à la version exacte du problème. Seul le testeur de langage régulier d'arbre apporte un contraste par rapport à celui du Théorème 2.2 précédemment énoncé, d'autant plus qu'il est possible de le rendre tolérant. Ce testeur est exponentiel par rapport à celui sur les mots car il utilise en sous-routine le testeur pour les langages algébriques.

THÉORÈME 2.15. *Pour tout réel $\varepsilon > 0$ et tout langage régulier L d'arbres d'arité 2 sur un alphabet fini Σ, il existe un $(\varepsilon^4, O(\varepsilon))$-testeur tolérant de L dont la complexité en requêtes est en $|\Sigma|^{O(1/\varepsilon^5)}$ et la complexité en temps est en $2^{|\Sigma|^{O(1/\varepsilon^5)}}$.*
De plus, étant donné un automate (non déterministe) d'arbres à m états reconnaissant L, le testeur peut être construit en temps exponentiel en $m^{|\Sigma|^{O(1/\varepsilon^5)}}$.

CHAPITRE 3

Algorithmique quantique $|\psi\rangle$

1. Préliminaires

Cette partie est une introduction au modèle du calcul quantique. Les notions mathématiques de la physique quantique nécessaires pour notre étude sont présentées comme axiomes. Pour une présentation plus détaillée on pourra consulter un des ouvrages [Pre, NC00, KSV02].

1.1. L'état quantique. Considérons l'ensemble S des états possibles d'un système classique. S peut être fini lorsque par exemple $S = \{0, 1\}$ pour les états d'un bit, ou $S = \{0, 1\}^n$ pour n bits, mais aussi infini pour les états possibles d'une machine de Turing. Puisque nous nous intéressons uniquement aux systèmes finis, S sera supposé fini et pour simplifier nous supposons $S = \{0, 1, \ldots, N-1\}$. Le formalisme qui suit servira à décrire les états possibles de n bits correspondant au cas $S = \{0, 1\}^n$, *i.e.* $N = 2^n$.

Afin d'amener progressivement le formalisme de l'évolution quantique, nous reprenons d'abord la description de l'évolution probabiliste d'un système d'états possibles S.

Le système est dans un *état probabiliste* lorsqu'il est décrit par une distribution de probabilité sur S, *i.e.* un vecteur $v \in \mathbb{R}^S$ à coordonnées positives ou nulles et de somme 1. Alors la *probabilité d'observer* le système dans un état $i \in S$ est donnée par la coordonnée v_i. Par exemple, un *bit probabiliste* est un élément $(p, 1-p) \in \mathbb{R}^2$, avec $0 \leq p \leq 1$. L'évolution d'un système probabiliste est modélisée par une matrice réelle A de taille N dont les coefficients sont positifs ou nuls et de somme 1 sur chacune des colonnes. Une telle matrice est dite *stochastique*. Après l'évolution A, l'état v se retrouve envoyé vers l'état $v' = Av$.

Lorsqu'un système est en *superposition quantique*, son *état quantique* est décrit par un vecteur normé de l'espace de Hilbert $(\mathbb{C}^S, \|\cdot\|)$. Dans la *notation de Dirac*, un tel vecteur est représenté par $|\psi\rangle$, son vecteur adjoint $|\psi\rangle^\dagger$ par $\langle\psi|$, le produit scalaire (resp. extérieur) entre $|\psi\rangle$ et $|\psi'\rangle$ par $\langle\psi|\psi'\rangle$ (resp. $|\psi\rangle\langle\psi'|$). Un *bit quantique*, ou *qubit*, est donc un élément $|b\rangle = \alpha|0\rangle + \beta|1\rangle$, avec $\alpha, \beta \in \mathbb{C}$ tels que $|\alpha|^2 + |\beta|^2 = 1$. L'évolution d'une superposition quantique est décrite par une matrice *unitaire* A de dimension N, *i.e.* qui préserve la norme, ou encore une matrice complexe telle que $A^\dagger A = \mathrm{Id}$. La *mesure de von Neumann* d'un système consiste à transformer son état quantique en un état probabiliste de sorte que la *probabilité d'observer* $|\psi\rangle$ dans un état $i \in S$ après la mesure, est donnée par le carré du module de la coordonné $|\psi\rangle_i$, soit $|\langle i|\psi\rangle|^2$. Implicitement cette mesure est reliée à la base canonique $(|i\rangle)_{i\in S}$ de $\mathbb{C}^S$. Une mesure de von Neumann dans une autre base orthonormée est tout à fait envisageable. Alors l'observation de $|\psi\rangle \in \mathbb{C}^S$ par une *mesure de von Neumann dans une base orthonormée* B de $\mathbb{C}^S$ produit avec probabilité $|\langle\phi|\psi\rangle|^2$ l'état $|\phi\rangle \in B$. La base canonique $(|i\rangle)_{i\in S}$ est appelée *base de calcul*. Si $S = S_1 \times S_2$ représente la corrélation de deux systèmes, alors l'état quantique du système global est décrit par un vecteur $|\psi\rangle \in \mathbb{C}^{S_1 \times S_2} = \mathbb{C}^{S_1} \otimes \mathbb{C}^{S_2}$. Les *états séparés* sont de la forme $|\psi_1\rangle \otimes |\psi_2\rangle$, et *enchevêtrés* sinon. Une évolution séparée sur S_1 et S_2 est alors de la forme $A_1 \otimes A_2$. Une mesure du premier système est possible : le nouvel état obtenu est à la projection orthogonale du vecteur d'état sur l'espace des états compatibles avec le résultat de la mesure. Il faut alors renormaliser l'état après sa projection.

1.2. Circuit quantique. Les circuits quantiques sont définis de manière analogue aux circuits logiques et plus précisément aux circuits réversibles, puisque toute transformation quantique est réversible puisqu'unitaire.

Une *porte* (quantique) G est une transformation unitaire n'agissant que sur un nombre constant de qubits, soit au plus 3 pour fixer les idées. Plus précisément, G peut s'écrire $U \otimes \mathrm{Id}$ où $U \in \mathcal{U}(2^3)$.

Par commodité, la matrice Id du produit tensoriel est la plupart du temps omise. Voici des exemples de portes importantes définies sur les vecteurs de base de calcul, pour $b = 0, 1$:

- Porte négation NOT : $\mathrm{NOT}|b\rangle = |1-b\rangle$;
- Porte Hadamard H : $\mathrm{H}|b\rangle = \frac{1}{\sqrt{2}}(|0\rangle + (-1)^b|1\rangle)$;
- Porte négation contrôlée c-NOT : $\text{c-NOT}|0b\rangle = |0b\rangle$ et $\text{c-NOT}|1b\rangle = |1\,(1-b)\rangle$;
- Porte de Toffoli T : $\mathrm{T}|0ab\rangle = |0ab\rangle$, $\mathrm{T}|a0b\rangle = |a0b\rangle$ et $\mathrm{T}|11b\rangle = |11\,(1-b)\rangle$;

Un *circuit* (quantique) est une suite de portes quantiques (voir Figure 3.1). Une transformation unitaire U se *décompose* en un circuit C, si le produit des portes de C donne U. Inversement, le circuit C *réalise* la transformation unitaire U. Une transformation unitaire U se *décompose à précision* ε en un circuit C, si le produit des portes de C donne une transformation unitaire V telle que $\|V - U\| \leq \varepsilon$, où $\|\cdot\|$ est la norme d'opérateur associée à la norme ℓ_2. La *taille* d'un circuit est le nombre de portes dont il est constitué.

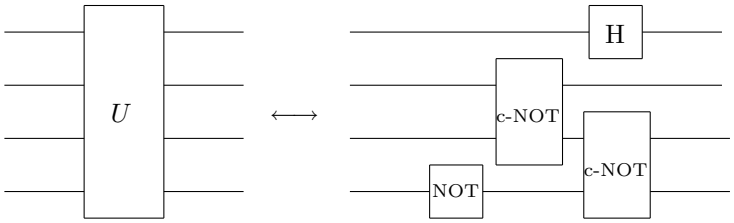

FIG. 3.1. Décomposition d'une transformation unitaire U en circuit de taille 4 $C = (H \otimes \mathrm{Id}_2 \otimes \text{c-NOT}) \cdot (\mathrm{Id}_2 \otimes \text{c-NOT} \otimes \mathrm{Id}_2) \cdot (\mathrm{Id}_8 \otimes \mathrm{NOT})$.

THÉORÈME 3.1. [BCC+95] *Toute transformation unitaire se décompose en portes sur* 1*-qubit et* c-NOT.

THÉORÈME 3.2. [Shi03] *Soit $\varepsilon > 0$ fixé. Toute transformation unitaire se décompose avec précision ε avec uniquement des portes parmi une des familles ci-dessous :*

- c-NOT *et* $\sqrt{\mathrm{H}}$*;*
- T *et* H.

Ce résultat nécessite deux commentaires. Tout d'abord contrairement aux circuits réversibles il existe des familles universelles de portes sur 2-qubit. Ensuite, il suffit d'ajouter à la porte de Toffoli, qui est universelle pour le calcul réversible, la porte de Hadamard pour obtenir une famille universelle pour le calcul quantique, à une précision donnée près.

Une des supériorités du calcul quantique est de pouvoir décomposer la transformée de Fourier sur un groupe abélien en un petit circuit quantique. La Figure 3.2 illustre ce résultat pour le groupe $(\mathbb{Z}_2)^n$.

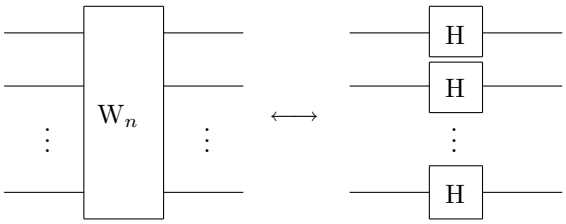

FIG. 3.2. La transformée de Walsh-Hadamard W_n, ou encore de Fourier sur le groupe $(\mathbb{Z}_2)^n$.

DÉFINITION 3.1. *Etant donnée une fonction $f : \{0,1\}^n \to \{0,1,\}^m$, un circuit quantique C* calcule *f s'il réalise une transformation unitaire U tel que pour tout $x \in \{0,1\}^n$, la mesure des m premiers qubits de $U|x,\bar{0}\rangle$ produit $f(x)$ avec probabilité au moins* 2/3*, i.e. :*

$$\forall x, \quad \sum_z |\langle f(x), z|U|x,\bar{0}\rangle|^2 \geq 2/3.$$

1.3. Algorithme quantique. Le passage des circuits quantiques aux machines de Turing quantiques se fait naturellement comme en calcul classique. Une machine de Turing déterministe sert à assurer l'uniformité des circuits.

DÉFINITION 3.2. *Une* machine de Turing quantique*, ou encore un* algorithme quantique*, est une machine de Turing déterministe A qui produit pour chaque entier n la description d'un circuit quantique C_n. La* complexité en temps *d'un algorithme quantique est la somme du temps qu'il faut pour décrire C_n et de la taille de C_n.*

La notion de calculabilité d'une fonction par un circuit quantique est naturellement étendue aux algorithmes quantiques.

DÉFINITION 3.3. *Un algorithme quantique* calcule *une fonction $f : \{0,1\}^* \to \{0,1\}^*$, si pour tout entier n le circuit C_n qu'il produit calcule la restriction f_n de f à $\{0,1\}^n$.*

2. Problèmes de groupe

2.1. Présentation. Les algorithmes de factorisation et de calcul du logarithme discret de Shor [Sho97] découlent de la résolution partielle d'un problème plus général qu'est le *problème du sous-groupe caché* (HIDDEN SUBGROUP) :

> HIDDEN SUBGROUP(G)
> *Entrée :* Une fonction f définie sur G qui *cache* un sous groupe inconnu $H \leq G$, *i.e.* constante dans chacune des classes de G/H, mais prenant des valeurs distinctes sur les classes de G/H.
> *Sortie :* Un ensemble de générateurs de H.

Intuitivement une telle fonction est alors non seulement H-périodique mais de plus injective à son sous-groupe de périodes près. Ce problème peut être résolu efficacement par un ordinateur quantique lorsque le groupe G est abélien (commutatif), fournissant entre autre un algorithme rapide pour la factorisation et le calcul du logarithme discret. Cet algorithme, s'il pouvait être exécuté par un ordinateur quantique, remettrait en cause une bonne partie de la cryptographie moderne (dont une présentation peut être obtenue dans le livre [Ste98]). Contrairement au cas abélien, très peu de résultats sont connus pour les groupes non abéliens, malgré le fait que de nombreux autres problèmes s'y réduisent.

L'un des exemples les plus importants est celui du problème de l'isomorphisme de graphes, qui se réduit au problème du sous-groupe caché lorsque G est le groupe symétrique S_n, qui n'est bien sûr pas abélien. Résoudre le problème de l'isomorphisme de graphes est un des défis actuels du calcul quantique. Une des voies pour y arriver consiste à résoudre le problème du sous-groupe caché pour de plus en plus de groupes non abéliens.

Nous avons à deux reprises contribué significativement dans cette voie. Une première fois [**IMS01**, **IMS03**], nous avons montré comment combiner les résultats de Shor [Sho97] avec ceux de la théorie des groupes boîtes noires (black-box groups). Nos techniques amènent de nouveaux résultats mais permettent aussi de redémontrer de manière directe, en les généralisant, des résultats antérieurs [Wat01] pour les groupes résolubles, et d'affiner l'étude des sous-groupes cachés distingués [HRT00].

Lorsque G est abélien, G est identifié à l'ensemble $\widehat{G}$ de ses caractères via un isomorphisme fixé $y \mapsto \chi_y$. L'*orthogonal de $H \leq G$* est défini par $H^\perp = \{y \in G : \forall h \in H, \chi_y(h) = 1\}$. La *transformée de Fourier quantique* sur G est la transformation unitaire définie pour tout $x \in G$ par

$$\mathrm{QFT}_G|x\rangle = \frac{1}{\sqrt{|G|}} \sum_{y \in G} \chi_y(x)|y\rangle$$

.

A l'aide de la transformée de Fourier quantique sur un groupe abélien G, il est possible de réaliser l'échantillonneur de Fourier quantique ci-dessous pour n'importe quelle fonction $f : G \to S$ donnée par un *oracle quantique*, *i.e.* une transformation unitaire U_f telle que $U_f|x\rangle|0\rangle = |x\rangle|f(x)\rangle$ (et éventuellement de son inverse $(U_f)^{-1}$). Un tel échantillonneur est en fait l'ingrédient principal pour résoudre HIDDEN SUBGROUP(G) lorsque G est abélien. Effectivement, si f cache un sous-groupe $H \leq G$, alors **Fourier sampling**$^f(G)$ génère une distribution uniforme d'éléments de $H^\perp$. A partir de cette distribution, H peut être retrouvé en temps probabiliste polynomial.

Fourier sampling$^f(G)$

(1) Créer l'état initial $|0\rangle_G|0\rangle$
(2) Appliquer QFT_G sur le premier registre
(3) Appeler la fonction f à l'aide de U_f
(4) Appliquer QFT_G sur le premier registre
(5) Retourner le résultat de la mesure du premier registre

2.2. Utilisation des travaux de Beals–Babai. Avant de citer un des résultats principaux de [BB93], rappelons qu'un *groupe boîte noire* est un groupe pour lequel les opérations de composition et d'inverse se font par oracle. La donnée du groupe proprement dite est faite par une liste de générateurs du groupe. L'encodage du groupe est un point important. En effet, un élément peut avoir plusieurs encodages possibles. Cette situation est même courante, quand par exemple on travaille sur les classes d'équivalence d'un groupe en utilisant l'encodage des éléments du groupe initial.

Beals et Babai définissent un paramètre $\nu(G)$ dépendant de la structure de G, mais dont la définition est plus que technique (voir [BB93]). En fait, il est suffisant de savoir que pour tout groupe résoluble G, le paramètre $\nu(G)$ vaut 1. De plus, ce paramètre est polynomial en $\log|G|$ pour une multitude de groupes importants, comme les groupes de permutation, ou les groupes finis de matrices sur des corps de nombres.

THÉORÈME 3.3. **(Beals–Babai** [BB93]**, Théorème 1.2)** *Soit G un groupe boîte noire fini avec un encodage non nécessairement unique. En supposant que sont donnés :*

(a) *l'ensemble des diviseurs premiers du cardinal $|G|$ de G,*
(b) *un oracle pour calculer des logarithmes discrets dans des corps de taille au plus $|G|$,*
(c) *un oracle constructif pour le test d'appartenance à un sous-groupe abélien élémentaire quelconque de G.*

Alors les tâches suivantes peuvent être résolues en temps probabiliste polynomial en $\log|G| + \nu(G)$:

(i) *tester l'appartenance à un sous-groupe quelconque de G,*
(ii) *calculer le cardinal de G ainsi qu'une présentation de G,*
(iii) *trouver des générateurs du centre de G,*
(iv) *construire une série de composition $G = G_1 \rhd G_2 \rhd \ldots \rhd G_t = 1$ pour G, avec des représentations des facteurs de compositions G_i/G_{i+1},*
(v) *trouver les sous-groupes de Sylow de G.*

Dans le contexte du calcul quantique, certaines tâches des hypothèses du Théorème 3.3 admettent des algorithmes quantiques efficaces. Ainsi, par les résultats de Shor [Sho97], l'oracle pour les calculs de logarithmes discrets peut être réalisé en temps quantique polynomial. Il en est de même pour les diviseurs premiers de $|G|$.

Quelques autres remarques sur l'algorithme sous-jacent au Théorème 3.3 permettent alors de le simplifier comme ci-dessous dans le cadre du calcul quantique.

COROLLAIRE 3.1. *Soit G un groupe boîte noire fini avec un encodage non nécessairement unique. En supposant que sont donnés :*

(a) *un oracle pour calculer l'ordre des éléments de G,*
(b) *un oracle constructif pour le test d'appartenance à un sous-groupe abélien élémentaire quelconque de G.*

Alors les tâches suivantes peuvent être résolues en temps quantique polynomial en $\log|G| + \nu(G)$:

(i) *test constructif d'appartenance à un sous-groupe quelconque de G,*
(ii) *–(v) identiques au Théorème 3.3.*

Dans le cas de l'encodage unique, ces hypothèses deviennent même efficacement réalisables par un algorithme quantique en utilisant les idées de Shor [Sho97]. Par conséquent le résultat suivant découle.

THÉORÈME 3.4. *Soit G un groupe boîte noire avec encodage unique. Alors chacune des tâches listées dans le Corollaire 3.1 est réalisable en temps quantique polynomial en* $\log|G| + \nu(G)$.

Ce résultat généralise une partie des résultats de Watrous [Wat01] pour les groupes résolubles.

Revenons maintenant au HIDDEN SUBGROUP. Nous sommes capables de le résoudre lorsque le sous-groupe est normal. Hallgren, Russell et Ta-Shma [HRT00] ont obtenu un résultat similaire, mais à la condition qu'une réalisation efficace de la transformée de Fourier quantique sur G existe. Cependant aucune réalisation existe actuellement pour le cas général, y compris les groupes résolubles. A l'opposé, notre algorithme ne réclame pas une telle hypothèse, mais sa complexité dépend du paramètre $\nu(G/N)$, qui, rappelons le, vaut 1 pour les groupes résolubles.

THÉORÈME 3.5. *Soit G un groupe boîte noire avec un encodage non nécessairement unique.* HIDDEN SUBGROUP*(G) restreint aux sous-groupes cachés distingués $N \leq G$ peut être résolu en temps quantique polynomial en* $\log|G| + \nu(G/N)$.

Pour illustrer le champ d'application de cette technique, voici un autre exemple de cas où HIDDEN SUBGROUP peut être résolu efficacement.

THÉORÈME 3.6. *Soit G un groupe boîte noire avec encodage unique.* HIDDEN SUBGROUP*(G) peut être résolu en temps quantique polynomial en* $\log|G| + |G'|$*, où G' est le sous-groupe dérivé de G.*

Ce résultat fournit un algorithme efficace lorsque G' est petit, comme pour le cas des p-groupes extra-spéciaux, où le sous-groupe dérivé coïncide avec son centre et $|G'| = p$.

D'autres résultats et idées de réduction avaient aussi été proposés, mais plus tard généralisés dans [**FIM$^+$03**] et détaillés ci-après.

2.3. Les groupes résolubles. Nous [**FIM$^+$03**] avons obtenu des résultats encourageants pour le cas des groupes résolubles de petits exposants. Ce résultat a été rendu possible par la combinaison de deux résultats indépendants. Le premier concerne le cas des groupes quasi-abéliens comme les produits semi-directs $G \rtimes \mathbb{Z}_2$, où G est un groupe abélien fini. Pour le groupe dihédral, *i.e.* $G = \mathbb{Z}_n$, ce problème a été résolu [EH00] en un nombre polynomial de requêtes à la fonction mais avec un post-traitement exponentiel. Nous avons pu résoudre ce problème en temps polynomial lorsque $G = (\mathbb{Z}_k)^n$, où k est une constante. Pour cela, nous avons dans la foulée élaboré un algorithme classique permettant de résoudre efficacement un système formé d'inéquations sur un corps fini. Ce résultat pourrait donc être d'un intérêt indépendant.

La première étape consiste à démontrer que HIDDEN SUBGROUP$(G \rtimes \mathbb{Z}_2)$ peut être résolu en temps quantique polynomial lorsque $G = \mathbb{Z}_p^n$, où $p > 2$ est un nombre premier fixé (le cas $p = 2$ se ramène en fait à une instance abélienne de HIDDEN SUBGROUP et peut donc se résoudre facilement). Pour simplifier, supposons qu'une fonction f cache un sous-groupe H de cardinal 2, et donc de la forme $H = \{(0,0),(1,u)\}$, où $u \in G$. Dans cette situation u est appelé la *translation cachée* de f, car les restrictions $f(0,\cdot)$ et $f(1,\cdot)$ sont injectives et de plus $f(0,x) = f(1,x+u)$ pour tout $x \in G$.

La partie quantique de l'algorithme consiste à oublier la structure non commutative du groupe, et à réaliser **Fourier sampling** sur le groupe abélien $\mathbb{Z}_p^n \times \mathbb{Z}_2$. Uniquement les échantillons de la forme $(y,1)$ seront utiles dans la suite. Ces éléments y ont la propriété importante de **ne pas** être orthogonaux à u. Ceci est cependant surprenant puisque, dans le cas abélien, les échantillons sont toujours orthogonaux au sous-groupe caché. Les propriétés de ces échantillons sont énoncées ci-dessous.

LEMME 3.1. *Soit G un groupe abélien. Soit f une fonction définie sur G et cachant une translation $u \neq 0$. Alors* ***Fourier sampling***$^f(G \times \mathbb{Z}_2)$ *renvoie un élément de $G \times \{1\}$ avec probabilité $1/2$. De plus, la probabilité d'échantilloner un élément $(y,1)$ ne dépend que de $\chi_y(u)$, et est 0 si et seulement si $y \in u^\perp$.*

Voici maintenant l'algorithme **Translation finding** qui trouve une translation cachée dans le cas de $G = \mathbb{Z}_p^n$. en temps quantique polynomial. L'idée principale est ici d'élever à la puissance $(p-1)$ chaque inéquation afin d'obtenir uniquement des équations. Ces équations restent linéaires sur le bon espace, puis une série d'arguments algébriques permettent de conclure.

Translation finding$^f(\mathbb{Z}_p^n)$

0. Si $f(0,0) = f(1,0)$ alors renvoyer 0
(1) $N \leftarrow 13p\binom{n+p-2}{p-1}$
(2) Pour $i = 1, \ldots, N$ faire
    $(z_i, b_i) \leftarrow$ **Fourier sampling**$^f(\mathbb{Z}_p^n \times \mathbb{Z}_2)$
(3) $\{y_1, \ldots, y_M\} \leftarrow \{z_i : b_i = 1\}$
(4) Pour $i = 1, \ldots, M$ faire $Y_i \leftarrow y_i^{(p-1)}$
(5) Résoudre le système d'équations linéaires
    $Y_1 \cdot U = 1, \ldots, Y_M \cdot U = 1$
(6) Si pas de solutions ou plus d'une solution, alors retourner `Echec`
(7) Soit $1 \leq j \leq n$ tel que le coefficient de x_j^{p-1} est 1 dans U
(8) Soit $v = (v_1, \ldots, v_n) \in \mathbb{Z}_p^n$ tel que $v_j = 1$ et v_k est la coordonnée de $x_k x_j^{p-2}$ dans U pour $k \neq j$
(9) Trouver $0 < a < p$ tel que $f_0(0) = f_1(av)$
(10) Retourner av

THÉORÈME 3.7. *Pour tout nombre premier $p > 2$, tout entier $n \geq 1$, et toute fonction $f : \mathbb{Z}_p^n \times Z_2 \to S$ cachant une translation, l'algorithme* **Translation Finding**$^f(\mathbb{Z}_p^n)$ *renvoie* `Echec` *avec probabilité au plus $1/2$, et sinon renvoie la translation de f. Le nombre d'évaluations de g est en $O(p(n+p)^{p-1})$, et sa complexité en temps en $(n+p)^{O(p)}$.*

La deuxième étape est la première récurrence rendue possible pour le problème du sous-groupe caché. Toutes les méthodes précédentes laissaient croire qu'un tel procédé récursif était impossible. Ici l'idée a été d'introduire le problème ORBIT COSET encore plus difficile que HIDDEN SUBGROUP mais dont la récursion est réalisable. ORBIT COSET(G) met en jeu une action quantique du groupe G, *i.e.* qui agit sur des ensembles d'états quantiques deux à deux orthogonaux.

ORBIT COSET(G)
Entrée : Deux états quantiques $|\phi_0\rangle$ et $|\phi_1\rangle$ sur lequel le groupe G agit.
Sortie : Rejeter si aucun élément de G n'envoie $|\phi_1\rangle$ sur $|\phi_0\rangle$.
Sinon, un ensemble de générateurs du sous-groupe stabilisateur de $|\phi_1\rangle$ et un élément de G qui envoie $|\phi_1\rangle$ sur $|\phi_0\rangle$ (par l'action de G).

Lorsque le groupe G est commutatif, le problème Orbit Coset sur G est en fait équivalent à celui du sous-groupe caché sur $G \rtimes \mathbb{Z}_2$. Nous avons montré qu'Orbit Coset sur G se réduit à l'Orbit Coset sur G/N et sur N, lorsque N est un sous-groupe distingué et résoluble de G. La preuve de ce résultat utilise une technique développée par Watrous [Wat01] pour résoudre des problèmes algorithmiques sur les groupes résolubles.

THÉORÈME 3.8. *Soit G un groupe boîte noire à encodage unique. Soit $N \lhd G, N \neq G$ un sous-groupe résoluble tel que N et G/N sont des sous-groupes boîtes noires à encodage unique. Alors* ORBIT COSET(G) *se réduit à* $\{$ORBIT COSET(K) : $K \leq N\}$ *et* ORBIT COSET(G/N) *en temps quantique polynomial.*

En prenant pour cas de base le résultat du Théorème 3.7, et en itérant un nombre constant de fois la récurrence du Théorème 3.8, nous avons résolu le problème pour les groupes résolubles d'exposants constants et dont la longueur de leur chaîne de dérivation est aussi constante.

THÉORÈME 3.9. *Soit G un groupe résoluble boîte noire avec encodage unique. Si G est d'exposant constant de longueur de chaîne de dérivation aussi constante, alors* ORBIT COSET(G) *peut être résolu en temps quantique polynomial.*
Si le sous-groupe dérivé G' de G est d'exposant constant et de longueur de chaîne de dérivation aussi constante, alors HIDDEN SUBGROUP(G) *peut être résolu en temps quantique polynomial.*

De plus, puisque selon [Gla89] tout groupe résoluble a une chaîne de dérivation de longueur en $O(\log \log |G|)$, ORBIT COSET et HIDDEN SUBGROUP admettent tout de même un algorithme quantique pseudo-polynomial pour tout groupe résoluble d'exposant constant.

THÉORÈME 3.10. *Soit G un groupe résoluble boîte noire avec encodage unique. Si G' est d'exposant constant, alors* HIDDEN SUBGROUP(G) *peut être résolu en temps quantique* $2^{(\log\log|G|)^2 \log\log\log|G|)}$.

Enfin, en appliquant la méthode de Kuperberg [Kup05] pour résoudre ORBIT COSET$(\mathbb{Z}_n)$ sur les facteurs de la chaîne de dérivation de G, il est possible d'obtenir un algorithme sous-exponentiel pour tout groupe résoluble.

THÉORÈME 3.11. *Soit G un groupe résoluble boîte noire avec encodage unique. Alors* HIDDEN SUBGROUP(G) *et* HIDDEN TRANSLATION(G) *peuvent être résolus en temps quantique* $2^{O(\sqrt{\log|G|}\cdot\log\log|G|)}$.

3. Problèmes de recherche

Gagnant en maturité, les algorithmes quantiques reposent maintenant sur de nouvelles techniques mieux calibrées. Nous avons contribué au développement de ces techniques, mais aussi à la recherche de nouveaux algorithmes construits à partir de ces techniques. Ces algorithmes suivent un nouveau courant. La plupart des applications de l'algorithme de factorisation de Shor [Sho97] ont été cernées. Force est de constater que les gains exponentiels des algorithmes quantiques concernent une classe de problèmes algébriques. Les gains pour les problèmes de nature plus combinatoire ne sont en général que polynomiaux. Ce résultat n'est pas surprenant puisqu'il a été montré [BBC$^+$01] que le gain d'un ordinateur quantique pour calculer une fonction totale ne pouvait être que polynomial.

Grover [Gro96] a donc ouvert un axe de recherche vers des algorithmes quantiques à gains polynomiaux dont les résultats sont surprenants. Plusieurs problèmes bien étudiés qui ont des complexités classiques similaires se retrouvent avec des complexités quantiques différentes. Nous croyons que le calcul quantique apporte ici un regard neuf sur la complexité.

3.1. Le problème des éléments distincts. Nous avons développé des résultats comparables à celui de l'algorithme de recherche de Grover [Gro96] en vue d'une application à ELEMENT DISTINCTNESS :

> ELEMENT DISTINCTNESS(n)
> *Entrée :* Une fonction f définie sur un ensemble de taille n
> *Output :* Rejeter si f est injective
> Sinon, une *collision*, *i.e.* une paire (x, y) telle que $x \neq y$ et $f(x) = f(y)$

Ce problème est important en cryptographie car la sécurité de nombreux protocoles repose sur la difficulté de trouver une telle paire de collision pour des fonctions dites de *hachage*.

Brassard, Høyer et Tapp [BHT97] ont aussi montré qu'il était possible de résoudre ELEMENT DISTINCTNESS en temps $\Theta((n/r)^{1/3})$ pour les fonctions définies sur un ensemble de taille n et de plus r-*vers*-1, pour $r \geq 2$, *i.e.* telles que pour chaque élément x, il existe $(r-1)$ éléments $y \neq x$ tels que $f(x) = f(y)$. On parle alors du problème COLLISION. Cet algorithme a été prouvé optimal par Shi [Shi02]. Rappelons que le meilleur algorithme probabiliste ne peut résoudre cette tâche qu'en temps $\Theta(\sqrt{n/r})$. Nous avons montré que cette tâche pouvait encore être résolue plus efficacement quantiquement que classiquement lorsque les collisions sont sans structure et en nombre indéterminé [**BDH$^+$01**, **BDH$^+$05**]. Effectivement, alors qu'un ordinateur probabiliste nécessite un temps $\Omega(n)$, il existe un algorithme quantique en temps $O(n^{3/4})$.

Nous verrons plus loin que ce résultat a été depuis amélioré par l'utilisation de l'équivalent quantique des marches aléatoires. Ce problème que nous avons mis en avant a contribué à aiguiser l'intérêt que la communauté porte maintenant aux marches quantiques.

3.2. Marches aléatoires. Les marches aléatoires constituent un outil formidable en algorithmique probabiliste. C'est donc tout naturellement, mais lentement, qu'elles ont été généralisées au calcul quantique. Ambainis [Amb04] a été le premier à construire un algorithme à base de marches *quantiques* qui a apporté un gain significatif. C'était pour ELEMENT DISTINCTNESS décrit ci-dessus.

Nous [**MSS05**, **MSS06**][**MN05**, **MN06**] avons développé deux algorithmes pour deux problèmes différents (voir les sections suivantes). Chacun de ces algorithmes utilise l'une des méthodes algorithmiques à base de marches quantiques proposées par Ambainis [Amb04] et par Szegedy [Sze04]. L'un des apports de ces travaux est de trouver une application à chacune de ces deux méthodes pour laquelle l'autre méthode échoue.

Nous [**MNRS07**] avons aussi nous-même développé une nouvelle méthode pour construire des algorithmes quantiques dont l'objectif est de trouver un élément marqué parmi les états d'une chaîne de Markov. Nous combinons un outil développé par Szegedy [Sze04] dans un but similaire, c'est-à-dire la *quantisation* d'une chaîne de Markov. L'usage que nous en faisons a une application plus vaste tout en utilisant des techniques plus simples issues de l'algorithme de Shor. C'est la première fois que de telles techniques sont ainsi couplées. Le résultat est que nous unifions et combinons plusieurs techniques [Amb04, Sze04] pour construire des algorithmes à base de marches quantiques.

3.2.1. *Deux algorithmes probabilistes très proches.* Pour une introduction aux chaînes de Markov et aux marches aléatoires on pourra consulter le livre [AF06] en préparation.

L'algorithme quantique optimal pour Element Distinctness découvert par Ambainis [Amb04] reformule le problème en terme de recherche d'un élément marqué dans un graphe de Johnson défini par le problème. Dans le graphe de Johnson, les sommets sont des sous-ensembles de taille fixée d'un ensemble lui aussi fixé. Les arêtes relient alors les sous-ensembles dont la différence symétrique est une constante donnée, en général 1. Dans ce dernier cas, les sous-ensembles reliés diffèrent exactement d'un élément. L'algorithme peut être vu comme une analogie quantique du processus de recherche suivant, où P est la matrice de transition d'une chaîne de Markov définie sur un espace d'états X.

Search Algorithm 1

(1) Initialiser x en échantillonnant X selon une distribution de probabilité s
(2) Répéter t_2 fois
  (a) Si l'état y atteint à l'étape précédente est marqué, alors retourner y et stopper
  (b) Sinon, simuler t_1 étapes de la chaîne de Markov P en partant de l'état courant y
(3) Si l'algorithme n'a pas déjà terminé, alors retourner 'aucun élément marqué' et stopper

Les paramètres t_1 et t_2 de l'algorithme sont déterminés par les propriétés de la chaîne de Markov P et de l'ensemble $M \subseteq X$ des éléments marqués. La distribution de probabilité s initiale peut être prise uniforme, égale à la distribution stationnaire de P, où encore concentré sur un état initial fixé. L'idée sous-jacente dans cet algorithme est illustrée en considérant le cas d'une chaîne de Markov P ergodique. Si t_1 est choisi suffisamment grand, alors l'état y de l'étape (2a) ci-dessus sera (approximativement) distribuée selon la distribution stationnaire de P. Par conséquent, la boucle extérieure effectue une série d'échantillonnages selon la distribution stationnaire jusqu'à ce qu'un élément marqué de M soit trouvé. Si t_2 est choisi inversement proportionnel à la probabilité qu'un état soit marqué selon la distribution stationnaire, alors l'algorithme trouvera avec succès un élément marqué avec grande probabilité.

L'analyse de la version quantique de cet algorithme par Ambainis dépend fortement de la structure des éléments marqués intervenant pour Element Distinctness. Elle n'est généralisable qu'à des problèmes de nature très proche, comme ceux que nous avons considérés pour la recherche d'un triangle dans un graphe (Triangle Finding) [**MSS05**, **MSS06**]. Inspiré par cet algorithme, Szegedy [Sze04] a alors conçu un algorithme quantique de recherche fonctionnant pour toute chaîne de Markov ergodique et symétrique. La distribution stationnaire est dans ce cas uniforme. L'algorithme de Szegedy peut être vu comme l'analogie quantique d'un processus classique subtilement différent du précédent mais plus naturel.

Search Algorithm 2

(1) Initialiser x en échantillonnant X selon une distribution de probabilité s
(2) Répéter t fois
  (a) Si l'état y atteint à l'étape précédente est marqué, alors retourner y et stopper
  (b) Sinon, simuler *une* étape de la chaîne de Markov P en partant de l'état courant y
(3) Si l'algorithme n'a pas déjà terminé, alors retourner 'aucun élément marqué' et stopper

Le paramètre t est maintenant déterminé à la fois par la chaîne de Markov et l'ensemble M des états marqués. Cet algorithme est une version glouton du premier : il vérifie après chaque étape de la chaîne

de Markov si un état marqué a été atteint, indépendamment du fait que la chaîne ait suffisamment mélangé ou non les états.

Etablissons maintenant formellement la complexité de ces deux algorithmes afin de clarifier leurs différences. Supposons que ces algorithmes maintiennent une structure de données d qui associe à chaque état $x \in X$ une donnée $d(x)$. La donnée $d(x)$ sert à décider si x est marqué ou non. Trois types de coûts différents apparaissent lors de la manipulation de d tout au long de ces algorithmes.

Setup cost S : Le coût pour échantillonner $x \in X$ suivant la distribution initiale s, et pour construire la structure de données $d(x)$ pour un état x.

Update cost U : Le coût pour simuler une transition de x vers y depuis un état $x \in X$ suivant la chaîne de Markov P, et pour mettre à jour $d(x)$ en $d(y)$.

Checking cost C : Le coût pour décider si $x \in M$ à l'aide de $d(x)$.

La notion de *coût* peut être vue comme un vecteur listant toutes les mesures de complexités étudiées par le problème considéré, telles que la complexité en temps, en espace ou en requêtes (nombre d'appels à un oracle). Nous pouvons maintenant énoncer des bornes sur l'efficacité des deux algorithmes précédents en termes des paramètres de coût ci-dessus. L'*écart spectral* d'une matrice à valeurs propres réelles est l'écart entre sa plus grande valeur propre et sa deuxième plus grande valeur propre.

THÉORÈME 3.12. *Soit $\delta > 0$ l'écart spectral d'une chaîne de Markov P ergodique et symétrique, et soit $\frac{|M|}{|X|} \geq \varepsilon > 0$ quand M est non vide. Si s est la distribution uniforme, alors*

(1) ***Search Algorithm 1*** *détermine s'il existe un élément marqué et en trouve un avec grande probabilité si $t_1 = \mathrm{O}(\frac{1}{\delta})$ et $t_2 = \mathrm{O}(\frac{1}{\varepsilon})$. Le coût total est alors en $\mathsf{S} + \frac{1}{\varepsilon}\left(\frac{1}{\delta}\mathsf{U} + \mathsf{C}\right)$.*

(2) ***Search Algorithm 2*** *détermine s'il existe un élément marqué et en trouve un avec grande probabilité si $t = \mathrm{O}(\frac{1}{\delta\varepsilon})$. Le coût total est alors en $\mathsf{S} + \frac{1}{\delta\varepsilon}\left(\mathsf{U} + \mathsf{C}\right)$.*

3.2.2. *Analogies quantiques.* Comme dans le cas probabiliste, le but des algorithmes quantiques de recherche est de trouver un élément marqué dans un ensemble fini X, à l'aide d'une structure de données d, qui est mise à jour tout au long de l'algorithme. Soit X_d l'ensemble des états associés à leurs données, *i.e.* $X_d = \{(x, d(x)) : x \in X\}$. Pour simplifier, la supposition est faite que $\bar{0} \in X$ ainsi que $d(\bar{0}) = \bar{0}$.

Les marches quantiques d'Ambainis et de Szegedy, et aussi la nôtre, peuvent être vues comme des marches sur les *arêtes* de la chaîne de Markov initiale, plutôt que sur les nœuds. Donc l'espace des états est un sous-espace de $\mathcal{H} = \mathbb{C}^{X \times X}$, ou $\mathcal{H}_d = \mathbb{C}^{X_d \times X_d}$ quand la structure de données est aussi incluse. Ainsi, à chaque étape, l'extrémité droite d'une arête (x, y) est 'mélangée' (de manière quantique) parmi les voisins de x, puis l'extrémité gauche parmi les voisins de la nouvelle extrémité droite. Cette modification n'est pas que technique, mais rendu nécessaire par la contrainte d'unitarité des évolutions quantiques. L'état initial de l'algorithme est explicitement relié à la distribution stationnaire π de P.

Comme pour le cas probabiliste, trois types de coût sont distingués.

(Quantum) Setup cost S : Le coût pour construire l'état $\sum_x \sqrt{\pi_x}|x\rangle_d|\bar{0}\rangle_d$

(Quantum) Update cost U : Le coût pour réaliser chacune des transformations unitaires suivantes et leurs inverses

$$|x\rangle_d|\bar{0}\rangle_d \quad \mapsto \quad |x\rangle_d \sum_y \sqrt{p_{xy}}|y\rangle_d, \qquad |\bar{0}\rangle_d|y\rangle_d \quad \mapsto \quad \sum_x \sqrt{p^*_{yx}}|x\rangle_d|y\rangle_d,$$

où $P^* = (p^*_{xy})$ est la chaîne de Markov à temps renversé, définie par les égalités $\pi_x p_{xy} = \pi_y p^*_{yx}$.

(Quantum) Checking cost C : Le coût pour réaliser la transformation unitaire qui envoie $|x\rangle_d|y\rangle_d$ vers $-|x\rangle_d|y\rangle_d$ si $x \in M$, et laisse l'état inchangé sinon.

Les algorithmes quantiques de recherche d'Ambainis et de Szegedy fournissent un gain quadratique sur les temps t_1, t_2 and t, par rapport à leurs versions probabilistes.

THÉORÈME 3.13 (Ambainis [Amb04]). *Soit P une marche aléatoire sur le graphe de Johnson sur des sous-ensembles de taille r d'un ensemble de taille m, où $r = \mathrm{o}(m)$. Soit M un ensemble vide, ou bien égal à la classe des sous-ensembles de taille r contenant un sous-ensemble fixe de taille $k \leq r$. Alors l'écart spectal de P $\delta \in \Omega(\frac{1}{r})$, la fraction des éléments marqués est 0 ou $\varepsilon = \Omega(\frac{r^k}{m^k})$. De plus, il existe un algorithme quantique qui détermine si M est non vide et trouve un élément de M avec grande probabilité, avec un coût en $\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C})$.*

Dans le cas de Element Distinctness avec une unique collision, $k = 2$ puisqu'une collision correspond aux deux éléments $x \neq y$ recherchés tels que $f(x) = f(y)$. Alors le choix $r = n^{2/3}$ et $m = n$, fournit l'algorithme optimal d'Ambainis en $O(n^{2/3})$.

Théorème 3.14 (Szegedy [Sze04]). *Soit $\delta > 0$ l'écart spectral d'une chaîne de Markov P ergodique et symétrique, et soit $\frac{|M|}{|X|} \geq \varepsilon > 0$ quand M est non vide. Alors, il existe un algorithme quantique qui détermine si M est non vide avec grande probabilité et un coût en $\mathsf{S} + \frac{1}{\sqrt{\delta\varepsilon}}(\mathsf{U} + \mathsf{C})$.*

Si le coût C est significativement plus grand que le coût U, alors l'approche d'Ambainis sera plus efficace. De plus, l'algorithme *trouvera* un élément marqué s'il en existe un. Cette efficacité est illustrée par notre algorithme pour Triangle Finding [**MSS05**, **MSS06**]. Cet algorithme utilise deux marches quantiques à la Ambainis de manière récursive, alors que l'approche de Szegedy semble être moins efficace pour ce problème. Néanmoins, l'approche de Szegedy a d'autres avantages car elle s'applique sur une classe plus grande de chaînes de Markov et pour des ensembles quelconques d'états marqués. En utilisant ces possibilités, nous avons construit un algorithme optimal (à un facteur logarithme près) pour le test de commutativité d'un groupe (Commutativity Testing) [**MN05**, **MN06**], et qui n'a aucun équivalent par l'approche d'Ambainis.

3.2.3. *Contributions.* Notre nouvelle approche [**MNRS07**] est l'analogie quantique de l'algorithme probabiliste **Search Algorithm 1**, qui fonctionne pour toute chaîne de Markov ergodique. Cependant, pour simplifier l'exposition de nos résultats, la chaîne P est supposée *réversible*, *i.e.* P coïncide avec sa chaîne à temps renversé P^* ou encore $\pi_x p_{xy} = \pi_y p_{yx}$.

Théorème 3.15. *Soient $\delta > 0$ l'écart spectral d'une chaîne de Markov ergodique et réversible, et $\varepsilon > 0$ un minorant de la probabilité qu'un élément choisi selon la distribution stationnaire π de P soit marqué, i.e. $\Pr_\pi(x \in M) \geq \varepsilon$, quand M est non vide. Alors il existe un algorithme quantique qui détermine s'il existe un élément marqué et en trouve un, avec grande probabilité et avec un coût en $\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C})$.*

Lorsque la chaîne P n'est plus réversible, alors ce théorème reste vrai si δ désigne l'écart entre les deux plus grandes valeurs singulières de la matrice $\operatorname{diag}(\pi)^{1/2} P \operatorname{diag}(\pi)^{-1/2}$.

Ce théorème étend donc considérablement le champ d'application des approches précédentes symbolisées par les Théorèmes 3.13 et 3.14. Il combine à la fois les bénéfices de ces deux approches en étant capable de toujours trouver un élément marqué s'il en existe, le tout avec le coût le plus petit des deux, tout en étant applicable à une classe de chaînes de Markov plus grande. De plus, cet algorithme est conceptuellement plus simple, évite les difficultés techniques des approches précédentes, et fournit des améliorations en plusieurs points aux algorithmes quantiques existant précédemment à base de marches quantiques. Plus précisément, nous donnons une méthode directe pour tout algorithme à la Ambainis dans le cas de solutions multiples, sans avoir besoin de réduire le problème au cas d'une solution unique comme Ambainis le faisait. Ceci s'applique pour Element Distinctness et Triangle Finding. Pour Element Distinctness, la vérification du produit de matrices [BŠ06], et Commutativity Testing, alors qu'un algorithme à la Szegedy détecte seulement la présence d'une solution (ou d'une erreur dans le cas du produit de matrices), notre approche trouve cette solution avec le même coût. Enfin, nous améliorons la complexité pour Triangle Finding en terme de nombre de requêtes au graphe de $\mathrm{O}(n^{1.3}\mathrm{polylog}(n))$ à $\mathrm{O}(n^{1.3})$.

3.3. Propriétés de graphe. La complexité en requêtes des propriétés de graphe est devenue célèbre classiquement à cause de la conjecture d'Aanderaa et Rosenberg qui prétend que n'importe quelle propriété monotone de graphe (*i.e.* invariante par permutation des sommets) non constante a une complexité déterministe en requêtes en $\binom{n}{2}$, où n est le nombre de nœuds du graphe. Dans ce modèle, une requête (i, j) consiste à interroger le graphe comme oracle pour savoir si la paire de sommets (i, j) est oui ou non une arête du graphe.

Si la conjecture est toujours ouverte, la borne inférieure en $\Omega(n^2)$ a été prouvée par Rivest et Vuillemen [RV76]. Pour le cas aléatoire, les bornes inférieures générales prouvées sont loin de la conjecture. La première borne inférieure non linéaire à été prouvée par Yao [Yao87], puis améliorée par Hajnal [Haj91] à $\Omega(n^{4/3})$, et enfin à $\Omega(n^{4/3} \log^{1/3} n)$ [CK01]. Dans le cas quantique, la question a été résolue pour des calculs quantiques sans erreur en prouvant la borne $\Omega(n^2)$ [BCWZ99]. Dans le cas plus pertinent du calcul avec erreur bornée, seul le résultat général $\Omega(n^{2/3} \log^{1/6} n)$ a été prouvé par Yao [Yao03].

Remarquable est la différence de situation dans le domaine quantique. En effet la plupart des propriétés connues ont une complexité en requêtes de l'odre de $\Theta(n)$, $\Theta(n^{3/2})$ ou encore $\Theta(n^2)$. La version quantique de la conjecture d'Aanderaa et Rosenberg est donc uniquement une borne inférieure $\Omega(n)$ plutôt qu'une caractérisation exacte.

De manière orthogonale, nous cherchons à cerner les propriétés de graphe dont la complexité en requêtes n'entre pas dans un des trois cas cités plus haut. Un bon candidat est la propriété pour un graphe de contenir un triangle. Nous [**MSS05**, **MSS06**] avons en effet présenté deux nouveaux algorithmes pour ce problème, et plus précisément pour sa version forte Triangle Finding(n), qui consiste à trouver un triangle s'il en existe un. Le premier utilise $O(n^{10/7})$ requêtes au graphe et $O(\log n)$ bits quantiques, et le deuxième $O(n^{13/10})$ requêtes au graphe mais $O(n)$ bits quantiques. Ces algorithmes sont actuellement les meilleurs, mais pourtant leurs complexités sont encore loin de la seule borne inférieure connue en $\Omega(n)$. Il est important de préciser que le premier algorithme n'utilise que des variantes de l'algorithme de Grover, et que le deuxième utilise une marche quantique. Cette marche a été introduite par Ambainis [Amb04] pour résoudre Element Distinctness. Nous avons généralisé cette approche pour l'appliquer à notre problème. Il est à noter que la technique de marches quantiques de Szegedy [Sze04] ne fonctionne pas pour ce problème. Nous dérivons aussi toute une série d'algorithmes pour des propriétés monotones de graphe, dont la complexité en requêtes ne dépend que de la complexité en certificat de la propriété. Nous avons ainsi montré l'utilisation prometteuse des marches aléatoires pour les propriétés de graphe.

3.4. Test de commutativité. Nous [**MN05**, **MN06**] avons considéré le problème du test de commutativité (Commutativity Testing(k)) d'un groupe donné par boîte noire et spécifié par k générateurs. La complexité du nombre d'opérations de groupe de ce problème en fonction de k a été d'abord considérée par Pak [Pak00], qui a donné un algorithme probabiliste en $O(k)$. Nous avons construit un algorithme quantique optimal dont la complexité est en $O(k^{2/3})$ à des facteurs logarithmiques près. Cet algorithme utilise la technique de quantisation de Szegedy [Sze04] d'une chaîne de Markov, alors que la marche d'Ambainis [Amb04] échoue ici. En prouvant que notre algorithme quantique est optimal, nous prouvons au passage que celui (probabiliste) de Pak l'est aussi.

CHAPITRE 4

Bornes inférieures $\models_\varepsilon$ / $|\psi\rangle$

1. OBDD approché $\models_\varepsilon$

Nous avons vu à la Section 3.1 qu'une abstraction probabiliste pouvait être utilisée pour réduire la taille de systèmes de transitions pour des programmes dont la spécification est liée à des propriétés de graphes telles que la bipartition. Rappelons que les OBDD sont une représentation couramment utilisée en model checking pour les systèmes de transitions et les spécifications. Intuitivement un OBDD (ordered binary decision diagram) est un arbre de décision dans lequel tous les n ?uds à même profondeur lisent la même seule variable sans pouvoir la relire ultérieurement.

Cette abstraction était nécessaire puisque la représentation en OBDD d'une propriété telle que la bipartition était déjà connue exponentielle [HMT88] ; toutefois, restait sans réponse de savoir si un OBDD pouvait être construit pour l'ε-bipartition, la version relâchée au sens du property testing de la bipartition pour un paramètre $\varepsilon > 0$. Si cela avait été le cas, les méthodes usuelles de vérification du model checking auraient été suffisantes pour décider l'ε-bipartition. Dans cette section, nous montrons que ce n'est pas le cas. Il s'agit à notre connaissance de la première borne inférieure en terme d'OBDD pour un problème de property testing.

1.1. Etape 1 : OBDD et complexité de la communication. La preuve qu'un OBDD pour la bipartition a nécessairement une taille exponentielle est basée sur la complexité de la communication.

Dans un jeu de complexité de communication, chacun des deux joueurs reçoit une entrée, le joueur I reçoit x et le joueur 2 reçoit y, et leur objectif est de calculer une fonction $f(x, y)$. Pour calculer f, les joueurs suivent un protocole, dans lequel chaque joueur envoie alternativement à l'autre joueur un message, et à la fin du protocole, l'un des deux joueurs renvoie la valeur de $f(x, y)$. La *complexité de communication* d'une fonction f est la taille totale de tous les messages échangés pour le pire cas des entrées dans le meilleur protocole. Dans ce modèle, aucune limite sur la puissance de calcul des deux joueurs n'est faite. Enfin, quand un seul message est envoyé du joueur I au joueur II, on parle alors de complexité de la communication *à sens unique*.

Obtenir un minorant de la complexité de communication d'une fonction suffit à minorer sa taille en terme d'OBDD. En effet, soit un OBDD de largeur w représentant une fonction f en examinant les variables composant l'entrée de f dans l'ordre $x_1, \ldots, x_n$. Nous allons construire un protocole de communication à sens unique qui calcule f en utilisant un unique message de $\log w$ bits. Dans ce protocole, le joueur I reçoit la moitié des variables, et le joueur II l'autre moitié. Le joueur I envoie au joueur II l'état de l'OBDD après que la moitié des variables ait été lue. Il peut calculer cet état car ces variables sont en sa possession. L'autre joueur peut alors compléter l'évaluation de l'OBDD à l'aide de ses variables. La taille du message envoyé doit permettre de pouvoir coder l'état de l'OBDD, or si celui-ci est de largeur w, son état peut être codé par un entier compris entre 0 et $w - 1$, et donc par un mot binaire à $\log w$ bits.

Par cet argument, tout minorant m sur la complexité de la communication à sens unique d'une fonction f donne un minorant en 2^m sur la largeur des OBDD calculant f, pourvu que ce minorant reste valable indépendamment de la répartition des variables de f entre les joueurs I et II. En effet, rien ne peut présupposer dans quel ordre ces variables seront lues par l'OBDD optimal calculant f.

Le problème de l'ε-bipartition qui nous intéresse est formalisé par la fonction booléenne partielle suivante.

Définition 4.1 (ε-bipartition). *Soit $\varepsilon > 0$ un réel. L'*ε-bipartition *sur V est une fonction partielle sur les graphes G de sommets V telle que :*

$$f(G) = \begin{cases} 1 & \textit{si } G \textit{ est biparti,} \\ 0 & \textit{si } G \textit{ est } \varepsilon\textit{-éloigné de tout graphe biparti,} \\ \perp \textit{ (non défini)} & \textit{sinon.} \end{cases}$$

Un OBDD *calcule* une fonction partielle f si ses évaluations coïncident avec celles de f là où f est définie. Dans la suite, nous donnerons les éléments pour prouver le théorème suivant.

THÉORÈME 4.1. *Pour tout $\varepsilon > 0$ suffisamment petit, tout OBDD qui calcule l'ε-bipartition sur n sommets a une largeur en $2^{\Omega(n)}$.*

1.2. Etape 2 : Réduction entre problèmes de communication. Lorsqu'une fonction f est totale, sa *matrice de communication* est la matrice M_f telle que $(M_f)_{x,y} = f(x,y)$. Pour les fonctions partielles, posons $(M_f)_{x,y} = \star$ lorsque $f(x,y) = \perp$. La complexité de communication à sens unique de f est de l'ordre de $\log l$, où l est le nombre de lignes de M_f distinctes de manière *non ambiguë*, *i.e.* si sur une colonne une ligne contient un 0 alors que l'autre contient un 1 (ce résultat découle de [KN97, Page 144]).

De manière informelle, un problème de communication se réduit à un autre si la matrice de communication du premier problème est une sous-matrice de celle du deuxième. La minoration de la complexité de communication de l'ε-bipartition procède en deux étapes. La première consiste à réduire un autre problème, half-set disjointness, à l'ε-bipartition. Cette réduction sera pour une répartition de variables fixée pour half-set disjointness, et quelconque pour l'ε-bipartition. La deuxième étape donne un minorant du nombre de lignes distinctes de manière non ambiguë pour ce nouveau problème.

DÉFINITION 4.2 (half-set disjointness). *Soit S un ensemble fini. Le problème* half-set disjointness *dans S est une fonction partielle g sur les sous-ensembles $T_1, T_2 \subset S$ de taille $\lfloor |S|/4 \rfloor$ telle que :*

$$g(T_1,T_2) = \begin{cases} 1 & \textit{si } T_1 \cap T_2 = \emptyset, \\ 0 & \textit{si } |T_1 \cap T_2| \geq |T_1|/2, \\ \perp & \textit{sinon.} \end{cases}$$

Pour le problème half-set disjointness, une seule répartition des variables est considérée : l'entrée du joueur 1 est (un encodage de) T_1, et celle du joueur 2 (un encodage de) T_2.

Inspirée de la démarche de Hajnal, Maass et Turán [HMT88], la réduction construit une classe de graphes avec beaucoup de triangles, auxquels ils faut retirer εn^2 arêtes pour qu'ils deviennent bipartis.

La construction qui suit est illustrée par la Figure 4.1. Fixons une répartition R, Y des arêtes entre les joueurs 1 et 2. Soient A, B, C une partition des sommets V telle que $|B| = \Theta(n)$. Soient T_1 et T_2 des sous-ensembles de B. Maintenant, $G = (V, E)$ est le graphe obtenu en ajoutant le graphe biparti complet entre A et C, les arêtes R du joueur I entre A et B, et les arêtes Y du joueur II entre C et B. Le sous-graphe $G^{T_1,T_2} = (V, E \cap ((T_1 \times A) \dot\cup (T_2 \times B) \dot\cup (A \times C)))$ est l'objet qui nous intéresse. Le nombre d'arêtes à supprimer pour que G^{T_1,T_2} devienne biparti est relié au nombre de triangles qu'il contient, lui-même relié à la taille de l'intersection de T_1 et T_2 par construction. Par conséquent, G^{T_1,T_2} est biparti si $T_1 \cap T_2 = \emptyset$, et $\Theta(|T_1 \cap T_2|)$-éloigné de tout graphe biparti sinon.

LEMME 4.1. *Soient $\varepsilon > 0$ un réel suffisamment petit, et n un entier suffisamment grand. Alors pour toute répartition R, Y entre deux joueurs des arêtes d'un graphe de sommets V de taille n, le problème half-set disjointness dans un ensemble S de taille $\Theta(n)$ se réduit à l'ε-bipartition dans V (pour la répartition R, Y).*

La preuve se conclut en comptant le nombre de lignes distinctes de manière non ambiguë dans la matrice de communication de half-set disjointness via un argument probabiliste de comptage (voir [HMRAR98] pour en savoir plus sur la méthode probabiliste en combinatoire).

LEMME 4.2. *Soit S un ensemble fini. La matrice de communication de half-set disjointness dans S a au moins $\lfloor 2^{|S|/64} \rfloor$ de lignes distinctes de manière non ambiguë.*

2. Complexité en requêtes $|\psi\rangle$

Parce que prouver des bornes inférieures en temps est difficile, la plupart des résultats d'impossibilité sont relativisés à un oracle. Ainsi en classique comme en quantique, le modèle de requêtes

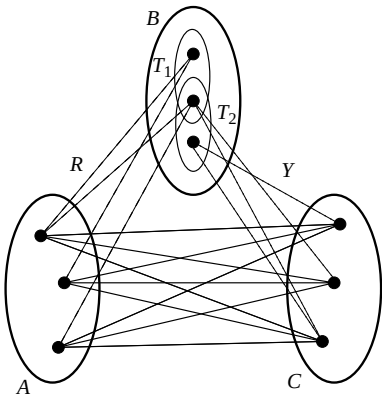

FIG. 4.1. Un exemple de G^{T_1,T_2} avec $|T_1 \cap T_2| \neq \emptyset$.

est souvent utilisé pour prouver des bornes inférieures concrètes de problèmes. En déterministe et en probabiliste, le terme d'arbre de décision est souvent employé. Dans le modèle de requêtes, l'accès à l'entrée du problème se fait par oracle. Si l'entrée est une chaîne booléenne, alors une question peut être une position, et la réponse la valeur booléenne à cette position dans la chaîne. Si l'entrée est un graphe, alors une question peut être une paire de sommets, et la réponse un booléen correspondant à l'existence d'une arête entre ces sommets. La *complexité en requêtes* d'un algorithme est le nombre de questions à l'oracle (codant l'entrée) pour le pire cas des entrées. La complexité en requêtes d'un problème est alors la plus petite des complexités en requêtes des algorithmes résolvant le problème.

Nous avons construit **[LM04**, **LM06]** une technique de preuve de bornes inférieures très générale pour à la fois la complexité en requêtes quantiques et classiques. Pour cela, nous avons introduit la complexité de Kolmogorov en tant qu'outil pour analyser la complexité de requêtes. La conséquence est une preuve novatrice en deux étapes. La première dépend du modèle, et la deuxième du problème. Cette structure simplifie à la fois la preuve et l'utilisation de la technique. Nos résultats généralisent toutes les techniques précédentes dites d'adversaires comme celles d'Ambainis [Amb02, Amb03a] et d'Aaronson [Aar04], ou encore de Barnum, Saks et Szegedy [BSS03] dont nous donnons une preuve plus simple qui ne passe pas par la notion de programmation semi-définie. En conséquence immédiate de notre technique, nous prouvons des bornes inférieures dépendant uniquement de la complexité en certificat de la fonction.

Depuis, il a été montré que toutes ces méthodes sont en fait équivalentes [ŠS06]. La preuve de ce résultat utilise explicitement notre généralisation des précédentes méthodes.

2.1. Résultat principal. La *complexité de Kolmogorov* (sans préfixe) $\mathsf{K}(a|b)$ est la longueur du plus petit programme qui retourne a étant donné b en entrée (étant donnée une Machine de Turing universelle fixée, ou encore un langage de programmation fixé). Notre résultat relie la complexité en requêtes d'un algorithme A qui calcule une fonction f aux quantités $\{\mathsf{K}(i|x,A), \mathsf{K}(i|y,A) : x_i \neq y_i\}$, pour chaque x, y tels que $f(x) \neq f(y)$.

THÉORÈME 4.2. *Il existe une constante $C > 0$ vérifiant ce qui suit. Soient Σ un ensemble fini, $n \geq 1$ un entier, et $S \subseteq \Sigma^n$ et S' deux ensembles. Soit $f : S \to S'$. Soit A un algorithme qui calcule $f(x)$ pour tout $x \in S$ avec une probabilité d'erreur bornée ε et une complexité en requêtes au plus T. Alors, pour chaque $x, y \in S$ tels que $f(x) \neq f(y)$:*

(1) *Si A est un algorithme quantique alors*

$$T \geq C \times \frac{1-2\sqrt{\varepsilon(1-\varepsilon)}}{\sum_{i:x_i \neq y_i} \sqrt{2^{-\mathsf{K}(i|x,A)-\mathsf{K}(i|y,A)}}};$$

(2) *Si A est un algorithme probabiliste alors*

$$T \geq C \times \frac{1-2\varepsilon}{\sum_{i:x_i \neq y_i} \min\left(2^{-\mathsf{K}(i|x,A)}, 2^{-\mathsf{K}(i|y,A)}\right)}.$$

Nous décrivons l'intuition de la preuve du Théorème 4.2. Considérons un algorithme A supposé calculer f avec une complexité en requêtes de T sur deux entrées x, y qui mènent vers deux réponses

différentes, *i.e.* $f(x) \neq f(y)$. D'une part, l'algorithme A doit interroger x et y à des positions où elles diffèrent avec probabilité moyenne au moins de l'ordre de $\frac{1}{T}$, ou alors A ne pourra pas distinguer x de y, et donc ne calculera pas la fonction f correctement. D'autre part, les questions qui sont posées avec une grande probabilité moyenne peuvent être décrites succinctement étant donné l'algorithme et l'entrée, en utilisant le code de Shannon-Fano. Si deux entrées x, y, telles que $f(x) \neq f(y)$, peuvent être exhibées de sorte que les positions où elles diffèrent n'admettent pas de descriptions succinctes, alors le nombre de requêtes doit être grand sur ces entrées.

Un fait remarquable est que ce raisonnement s'applique tout aussi bien au calcul probabiliste que quantique ; la seule différence est la rapidité avec laquelle les états de l'algorithme A correspondant à deux entrées différentes x et y peuvent diverger l'un de l'autre, afin de fournir deux réponses différentes.

Voici maintenant une application simple de cette méthode pour le problème de la recherche d'un 1 dans une chaîne de n bits, appelée *recherche de Grover* en raison de l'algorithme quantique de Grover en $O(\sqrt{n})$ pour ce problème. Ce problème est une abstraction du problème de la recherche d'un élément dans un tableau non trié de taille n

Fixons n et un algorithme A pour la recherche de Grover sur les entrées de n bits. Soit z une chaîne binaire de taille $\log n$ telle que $\mathsf{K}(z|n, A) \geq \log n$. L'existence d'une telle chaîne, dite *incompressible*, est garantie par la complexité de Kolmogorov. Soit i l'entier entre 0 et $n-1$ dont la représentation binaire est z. Considérons les entrées de n bits x, partout à 0, et y partout à 0 sauf à la position i qui est à 1. Alors puisque les chaînes i et z se déduisent l'une de l'autre, tout comme n et x, il vient, à une constante additive près, que $\mathsf{K}(i|x, A) \geq \log n$. De plus on a toujours $\mathsf{K}(i|y, A) \geq 0$. Par conséquent, la complexité en requêtes de A est au moins $\Omega(n)$ si A est un algorithme probabiliste, et au moins $\Omega(\sqrt{n})$ si A est un algorithme quantique.

2.2. Application à la méthode spectrale. Comme nous l'avons dit, toutes les méthodes par adversaires antérieures à nos travaux peuvent être simplement retrouvées en tant que corollaire du Théorème 4.2. Pour illustrer ces autres méthodes, nous énonçons la méthode spectrale de Barnum, Saks ans Szegedy [BSS03] initialement démontrée uniquement pour les fonctions booléennes, et que nous avons au passage généralisée à toute fonction. Cette généralisation a été entre autre utilisée par Santha et Szegedy [SS04]. Pour une matrice Γ, notons $\lambda(\Gamma)$ la plus grande valeur propre de Γ.

THÉORÈME 4.3. *Soient Σ un ensemble fini, $n \geq 1$ un entier, et $S \subseteq \Sigma^n$ et S' deux ensembles. Soit $f : S \to S'$. Soit Γ une matrice $S \times S$ symétrique à coefficients positifs ou nuls qui satisfait $\Gamma(x, y) = 0$ quand $f(x) = f(y)$. Pour $i = 1, \ldots, n$ soit Γ_i la matrice :*

$$\Gamma_i(x, y) = \begin{cases} 0, & \text{si } x_i = y_i \text{ ;} \\ \Gamma(x, y), & \text{sinon.} \end{cases}$$

Alors la complexité de requêtes de f pour un algorithme quantique est en

$$\Omega\left(\frac{\lambda(\Gamma)}{\max_i \lambda(\Gamma_i)}\right).$$

Un exemple relativement simple d'application de ce résultat est pour le problème de la recherche dichotomique. Les entrées $x \in \{1, 2, \ldots, n\}$ codent la position où se trouve la donnée recherchée dans un tableau de taille n. Les requêtes sont des requêtes de comparaisons, et donc la notation x_i désigne 0 si $i \leq x$ et 1 si $i > x$. La matrice Γ est alors définie par $\Gamma(x, y) = 1/|x - y|$ pour $x \neq y$ et $\Gamma(x, x) = 0$. Il est clair que $\lambda(\Gamma) = \Theta(\log n)$, alors qu'un peu de calcul permet de montrer que $\lambda(\Gamma_i) = \Theta(1)$. Il en résulte donc que les algorithmes quantique ne sont pas meilleurs que leurs analogues déterministes pour effectuer une recherche dans un tableau trié, puisqu'il leur faut aussi de l'ordre de $\Theta(\log n)$ comparaisons. Ce résultat peut aussi être généralisé au tri d'un tableau, et donne alors $\Theta(n \log n)$ comparaisons, comme le meilleur algorithme déterministe de tri.

2.3. Limitation de la méthode en terme de complexité de certificats. Soit f une fonction à valeurs booléennes. Pour toute instance positive $x \in \Sigma^n$ de f ($f(x)$=1), un *certificat positif* de f en x est un plus petit (en terme de taille) sous-ensemble $I \subseteq [n]$ d'indices de positions des symboles de x, tel que tout y satisfaisant $x_i = y_i$, pour chaque $i \in I$, satisfait aussi $f(y)$=1.

La *complexité de* 1*-certificat* de f, notée $C_1(f)$, est la taille du plus grand certificat positif de f en x, lorsque x parcourt toutes les instances positives x. La *complexité de* 0*-certificat* est définie de manière similaire pour les instances négatives x de f ($f(x) = 0$).

Le théorème suivant, dû à Troy Lee, est une conséquence relativement simple de notre méthode. Il prouve que notre méthode d'adversaires, ainsi que toutes celles que nous généralisons, ne peuvent prouver des minorants sur la complexité quantique en requêtes plus grande que $\min(\sqrt{nC_0(f)}, \sqrt{nC_1(f)})$, pour des fonctions f arbitraires. Indépendamment, Szegedy [Sze03] a montré le même résultat pour la méthode de [BSS03], et Zhang [Zha05] pour la méthode de [Amb03b]. Zhang a aussi montré une borne en $\sqrt{C_0(f)C_1(f)}$, pour les fonctions f totales, qui peut aussi se démontrer simplement avec notre méthode.

THÉORÈME 4.4. *Soient Σ un ensemble fini, $n \geq 1$ un entier, et $S \subseteq \Sigma^n$ un ensemble. Soit $f : S \to \{0,1\}$. Alors tout minorant de la complexité en requêtes d'un algorithme quantique calculant f donné par le Théorème 4.2 est en $O(\min(\sqrt{nC_0(f)}, \sqrt{nC_1(f)}))$.*

2.4. Application à la connexité d'un graphe. Pour conclure cette partie, nous montrons comment démontrer un résultat de [DHHM04]. Dans ce résultat, l'accès au graphe est effectué par requêtes à sa matrice d'adjacence. Les graphes sont non-orientés. A une question (i, j), l'oracle renvoie 1 si l'arête (i, j) existe, et 0 sinon.

THÉORÈME 4.5 ([DHHM04]). *La complexité en requêtes de tout algorithme quantique décidant si un graphe à n sommets est connexe est en $\Omega(n^{3/2})$.*

La preuve consiste à construire un graphe connexe G et un autre non-connexe H en utilisant une chaîne incompressible au sens de la complexité de Kolmogorov. Soit S une telle chaîne de longueur $\log(n-1)! + \log\binom{n}{2}$, coupées en deux parties S_1 and S_2 de longueurs respectives $\log(n-1)!$ et $\log\binom{n}{2}$. L'interprétation de S_1 (voir Figure 4.2) est la représentation d'un cycle hamiltonien $C = (\pi(0), \pi(1) \cdots \pi(n-1), \pi(0))$ parcourant n sommets, où π est une permutation de $\{0, 1, \ldots, n-1\}$ telle que $\pi(0) = 0$. Soit G le graphe formé du cycle C, tel que $\mathsf{K}(G) = \mathsf{K}(\pi)$. Maintenant, S_2 (voir Figure 4.2) peut être interprété comme la représentation d'une paire de sommets distincts s, t. Soit alors H obtenu depuis G en cassant le cycle en deux aux sommets s et t, et en reformant deux cycles, *i.e.* $H = G \setminus \{(\pi(s), \pi(s+1)), (\pi(t), \pi(t+1))\} \cup \{(\pi(s), \pi(t+1)), (\pi(s+1), \pi(t))\}$, où l'addition est modulo n.

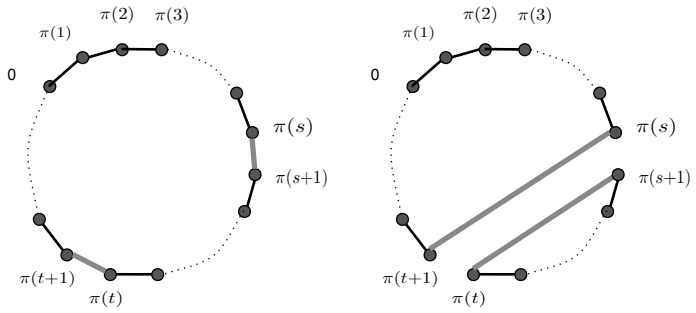

FIG. 4.2. Construction des graphes G et H.

Pour les quatre arêtes e où G et H diffèrent, nous allons montrer que $\mathsf{K}(e|G) + \mathsf{K}(e|H) \geq 3\log n - 4$, ce qui prouve le théorème à l'aide du Théorème 4.2.

Soient e_-, e'_- les arêtes retirées de G, et e_+, e'_+ celles ajoutées à G. A une constante additive près, il est clair que $\mathsf{K}(e_+|G) = \mathsf{K}(e'_+|G)$ et $\mathsf{K}(e_-|H) = \mathsf{K}(e'_-|H)$.

Supposons sans perte de généralités, que $e_- = (\pi(s), \pi(s+1))$, et que le plus petit cycle de H contient $\pi(s)$. Soit $l \leq n/2$ la longueur de ce cycle. Puisque $\mathsf{K}(s|G) = \mathsf{K}(e_-|G)$, et $\mathsf{K}(e_-|H) = \mathsf{K}(\pi, s, t|H)$, il vient que

$$\begin{aligned}
\log(n-1)! + \log\binom{n}{2} &\leq \mathsf{K}(S) \\
&\leq \mathsf{K}(G) + \mathsf{K}(s|G) + \mathsf{K}(t|G) \\
&\leq \mathsf{K}(G) + \mathsf{K}(e_-|G) + \log n \\
\mathsf{K}(e_-|G) &\geq \log\binom{n}{2} - \log n = \log\tfrac{n-1}{2}.
\end{aligned}$$

De plus,

$$\begin{aligned}\mathsf{K}(H) &\leq \mathsf{K}(l) + \log \tfrac{(n-1)!}{(n-l)!} + \log(n-l-1)! \\ &\leq \log(\tfrac{n}{2}) + \log(n-1)! - \log(n-l) \\ &\leq \log(n-1)!\,.\end{aligned}$$

Donc,

$$\begin{aligned}\log(n-1)! + \log\binom{n}{2} &\leq \mathsf{K}(S) \\ &\leq \mathsf{K}(H) + \mathsf{K}(\pi, s, t|H) \\ &\leq \mathsf{K}(H) + \mathsf{K}(e_-|H) \\ &\leq \log(n-1)! + \mathsf{K}(e_-|H) \\ \mathsf{K}(e_-|H) &\geq \log\binom{n}{2}.\end{aligned}$$

Pour les arêtes ajoutées, e_+, e'_+, supposons sans perte de généralités que $e_+ = (\pi(s), \pi(t+1))$. Puisque S est incompressible, $\mathsf{K}(e_+|G) = \mathsf{K}(s,t|G) \geq \log\binom{n}{2}$. De plus, $\mathsf{K}(S) \leq \mathsf{K}(H) + \mathsf{K}(e_+|H) + \mathsf{K}(e'_+|H)$, et $\mathsf{K}(e'_+|H) \leq \log n$, donc $\mathsf{K}(e_+|H) \geq \log\binom{n}{2} - \log n = \log\frac{n-1}{2}$. Ensuite, les mêmes arguments appliqués à e'_+ montrent de manière similaire que $\mathsf{K}(e'_+|H) \geq \log\frac{n-1}{2}$, et le résultat en découle.

CHAPITRE 5

Vérification quantique $\models_\varepsilon + |\psi\rangle$

Deux extensions du test pour le calcul quantique ont été étudiées. La première approche considère le test de fonctions déterministes par une procédure quantique. La seconde approche considère le test de dispositifs quantiques par des procédures classiques. Il s'agit donc d'auto-test.

1. Vérification approchée à l'aide d'un ordinateur quantique $\models_\varepsilon^{|\psi\rangle}$

Cette nouvelle approche, à laquelle nous avons contribué, concerne le test de fonctions déterministes par une procédure quantique. Il s'agit du quantum property testing introduit par Buhrman, Fortnow, Newman, and Röhrig [BFNR03]. Leur principale contribution est d'avoir mis en évidence une propriété difficile à décider quantiquement et difficile à tester classiquement, alors qu'elle est efficacement testable quantiquement. Ainsi le quantum property testing apporte non seulement un nouvel élément de la supériorité du calcul quantique sur le calcul classique, mais il peut aussi aider à résoudre des problèmes de décision difficiles pour le calcul quantique.

Nous [**FMSS03**] avons prolongé ces travaux en construisant des testeurs quantiques efficaces pour plusieurs *propriétés de groupe caché*, *i.e.* des propriétés de groupe de type $\exists$ et dont le problème de décision associé est exponentiellement difficile pour le calcul quantique. Pour cela, nous avons introduit une nouvelle technique dans l'analyse des testeurs quantiques qui nous permet de tester deux propriétés plus générales que les précédentes, et de manière encore plus efficace. Ces améliorations ont été rendues possibles par une analyse plus fine de nos testeurs. Nous avons en effet raffiné la méthode habituellement utilisée en test classique, qui consiste à montrer que lorsqu'une fonction f passe le test, elle peut être corrigée en une autre qui est proche et qui satisfait la propriété. La nouveauté est que cette correction n'est plus faite directement : la fonction est d'abord corrigée par une fonction probabiliste, puis cette dernière l'est par une autre déterministe.

Un de nos testeurs quantiques généralise les testeurs de périodicité étudiés dans [HH00, BFNR03]. Pour tout groupe fini G et tout sous-groupe distingué K, une fonction satisfait la propriété LARGER-PERIOD(K) s'il existe un sous-groupe distingué $H > K$ pour lequel f est H-périodique, *i.e.* $f(xh) = f(x)$ pour tout $x \in G$ et $h \in H$. Nous construisons un ε-testeur quantique pour cette propriété lorsque G est abélien.

Test Larger period$^f(G, K, \varepsilon)$

(1) $N \leftarrow 4\log(|G|)/\varepsilon$
(2) Pour $i = 1, \ldots, N$ échantillonner $y_i \leftarrow$ **Fourier sampling**$^f(G)$
(3) Accepter si et seulement si le groupe généré par $(y_i)_{1 \le i \le N}$ est strictement inclus dans $K^\perp$

THÉORÈME 5.1. *Pour tout groupe abélien G, tout sous-groupe K, et tout $0 < \varepsilon < 1$,* ***Test Larger period***(G, K, ε) *est un ε-testeur pour* LARGER-PERIOD(K) *de complexité en requêtes $O(\log(|G|)/\varepsilon)$ et de complexité en temps $(\log(|G|)/\varepsilon)^{O(1)}$.*

Ce résultat généralise les précédents résultats sous trois aspects :

(1) Le testeur fonctionne pour tout groupe abélien, alors que seuls les cas $G = \mathbb{Z}_n$ [HH00] et $G = \mathbb{Z}_2^n$ [BFNR03] avaient été considérés précédemment.
(2) La propriété testée dépend d'un paramètre K, alors que seul le cas $K = \{0\}$ avait été envisagé auparavant.

(3) La complexité en requêtes est seulement linéaire en $1/\varepsilon$ alors qu'elle était quadratique dans les travaux précédents.

Nous voyons que l'ingrédient principal de **Test Larger period** est la procédure **Fourier sampling**. Cette procédure reste un outil efficace pour HIDDEN SUBGROUP pour les groupes non abéliens lorsque le sous-groupe caché est distingué [HRT00, GSVV01]. Cependant, aucune réalisation efficace en termes de taille de circuit quantique n'est encore connue de **Fourier sampling** dans le cas général. Par conséquent, la complexité en requêtes des testeurs pour les groupes non abéliens est uniquement explicitée dans la généralisation suivante.

THÉORÈME 5.2. *Pour tout groupe fini G et sous-groupe distingué K et tout $0 < \varepsilon < 1$, il existe un ε-testeur pour* LARGER-PERIOD(K) *de complexité en requêtes* $O(\log(|G|)/\varepsilon)$.

2. Test de dispositifs quantiques $|\psi\rangle \models_\varepsilon$

Concernant l'auto-test quantique, Mayers et Yao [MY98] ont décrit la manière de tester si une source de photons était suffisamment fiable pour être utilisée dans le protocole de distribution quantique de clés secrètes de Bennett et Brassard [BB84]. Plus précisément, si la source passe ce test, alors la sécurité du protocole est assurée. Ce test se base sur le paradoxe Einstein-Podolsky-Rosen (EPR) [EPR35], et plus précisément sur la violation des inégalités de Bell [Bel64]. Intuitivement, ils montrent que si un état quantique maximise la violation des inégalités de Bell, alors nécessairement il s'agit d'un état EPR. Rappelons que la violation des inégalités de Bell prouve que le monde est quantique. La première mise en expérimentation de ces violations a été réalisée par l'équipe d'Aspect [AGR82, ADR82] entre 1980 et 1982 à l'Institut d'Optique situé à Orsay.

Nous [**DMMS00**, **DMMS07**] nous étions intéressés à l'auto-test des portes quantiques. Nous avions montré qu'il était possible de tester *classiquement* des portes quantiques en construisant la première famille de tests classiques qui permet d'estimer la fiabilité de portes quantiques. L'aspect original de ces auto-testeurs est que la porte testée est l'unique composante quantique utilisée. Le test peut uniquement générer des états classiques, appliquer la porte testée, puis mesurer la sortie dans la base des états classiques.

Cependant il restait un écueil. Non seulement la génération des états classiques et les mesures correspondantes étaient supposées fiables, mais une hypothèse forte sur la dimension du système était faite. Enfin, ce test était plus adapté aux portes qu'aux circuits.

Nous [**MMMO06**] venons de franchir une étape supplémentaire suite à une collaboration avec Mayers du California Institute of Technology et Mosca de l'Université de Waterloo (Canada) afin de combiner nos travaux sur l'auto-test quantique. Le résultat est un auto-testeur qui ne fait confiance ni aux appareils de mesure, ni aux sources d'états classiques et quantiques, ni à la dimension du système physique. Plus encore nous pouvons désormais tester un circuit tout entier. Ce travail est d'un intérêt à la fois théorique et pratique, car motivé par les récentes implémentations de dispositifs quantiques à base de RMN pour lesquels mêmes les états classiques sont difficiles à préparer.

2.1. Self-testing version génie logiciel. Nous rappelons la notion d'auto-test pour une machine, ou objet, quelconque. Cette approche nous permet d'unifier différentes situations d'auto-test. L'auto-testeur pour un objet d'une classe donnée est un algorithme (ou encore une machine de Turing) probabiliste à oracle. L'oracle est construit à partir de l'objet à tester selon une *interface* simulant les expériences dictées par l'auto-testeur et retournant leurs résultats. Intuitivement l'interface doit non seulement être plus simple que l'objet testé, mais doit aussi être complètement fiable. De plus l'objet testé est une boîte noire dont le contenu est invisible. L'oracle obtenu peut être contrairement à l'accoutumée probabiliste. Si l'objet correspond à une machine probabiliste ou quantique, alors l'observation du résultat d'une expérience ne peut être que probabiliste. Par contre si l'objet est une machine déterministe, l'oracle l'est aussi.

Enonçons la définition de l'auto-testeur pour tout type d'objet via une interface. Même s'il n'en est pas fait mention dans la définition, le but est d'obtenir un test plus simple que l'objet testé. Dans le cas d'un programme, un critère de simplicité a été formalisé par Blum et Kannan [BK95] sous le nom de (*little-oh property*). Dans le cadre du test de dispositifs quantiques, nous reviendrons sur ce point dans la section suivante.

DÉFINITION 5.1 (Auto-testeur). *Soient $\mathcal{C}$ la classe des objets à tester, et $\mathcal{F} \subseteq \mathcal{C}$ une propriété sur ces objets. Supposons $\mathcal{C}$ muni d'une interface $\mathcal{O}$. Soient* dist *une distance sur $\mathcal{C}$, et $\varepsilon_1, \varepsilon_2 \geq 0$. Un* $(\varepsilon_1, \varepsilon_2)$-auto-testeur *de $\mathcal{F}$ sur $\mathcal{C}$ avec l'interface $\mathcal{O}$ est un algorithme probabiliste T à oracle tel que pour tout $f \in \mathcal{C}$:*

- *si* $\mathsf{dist}(f, \mathcal{F}) \leq \varepsilon_1$, *alors* $T^{\mathcal{O}[f]}(\gamma)$ *accepte avec probabilité au moins* $2/3$;
- *si* $\mathsf{dist}(f, \mathcal{F}) > \varepsilon_2$, *alors* $T^{\mathcal{O}[f]}(\gamma)$ *rejette avec probabilité au moins* $2/3$.

Remarque. $T^{\mathcal{O}[f]}$ désigne l'algorithme T avec l'oracle probabiliste $\mathcal{O}[f]$ associé à f par $\mathcal{O}$.

2.2. Modélisation. Intuitivement l'interface quantique est un expérimentateur qui réalise des expériences simples. Ces expériences consistent à construire un circuit dont les composants sont uniquement ceux à tester, de mettre en entrée du circuit un état donné, et enfin d'observer la sortie du circuit selon un appareil de mesure fourni.

Les objets testés seront des transformations physiquement réalisables au sens de la mécanique quantique, communément appelées CPSO (completely positive superoperators). Intuitivement, un CPSO n'est rien d'autre qu'une transformation unitaire agissant sur un système plus grand que celui auquel l'observateur accède.

Aussi ne pas supposer connaître, ou contrôler, totalement l'espace où agit un CPSO autorise à supposer ce dernier unitaire. Bien sûr la réciproque est fausse. Voici les deux hypothèses considérées dans nos travaux.

(1) Connaître l'espace (et donc sa dimension) sur lequel agit G [**DMMS00**, **DMMS07**] (travail de thèse).

(2) Supposer que G est une transformation unitaire qui agit sur un espace dont on ne connaît qu'un sous-espace (et dont la dimension est donc inconnue) [**MMMO06**].

Le contexte du premier travail est moins général que le deuxième, et nous verrons dans la section suivante, qu'il est impossible de le généraliser au deuxième contexte. Par la suite, on se placera uniquement dans le deuxième contexte correspondant au travail [**MMMO06**] décrit dans cette partie.

2.3. Une conspiration contre le test de la porte Hadamard. Voici tout d'abord le testeur construit dans [**DMMS00**, **DMMS07**] afin de vérifier qu'un CPSO réalise une porte Hadamard (à une phase près sur l'état $|1\rangle$). Ce testeur est à comprendre dans un contexte où seuls les états $|0\rangle$ et $|1\rangle$ peuvent être préparés, seule la mesure dans la base de calcul peut être effectuée, et enfin seul le CPSO testé peut être appliqué autant de fois que voulu. Ce test consiste essentiellement à vérifier les statistiques correspondant aux expériences attendues suivantes qu'il répète suffisamment de fois pour les approcher avec une bonne précision (voir Figure 5.1) :

(1) La mesure de la sortie d'un CPSO G appliqué à $|0\rangle$ ou $|1\rangle$ fournit une distribution $50\% - 50\%$ de '0' et de '1'.

(2) La mesure de la sortie résultant de deux applications de G sur l'état $|0\rangle$ fourni 100% de '0'.

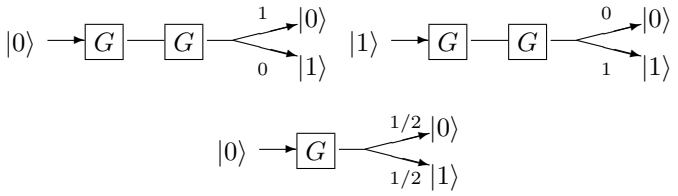

FIG. 5.1. Le test de la porte Hadamard.

Voici maintenant une conspiration très simple due à Wim van Dam. Elle consiste à montrer que si la dimension du système est inconnue, alors il existe un système expérimental qui passe le test précédent mais qui ne correspond pas à la porte Hadamard.

Le système du qubit est en fait encodé dans un système à 4 états, correspondant à 2 qubits. L'état supposé $|0\rangle$ est encodé en $|00\rangle$, et l'état supposé $|1\rangle$ en $|11\rangle$. Le CPSO supposé réaliser la porte Hadamard réalise en fait $|00\rangle \mapsto |01\rangle$, $|01\rangle \mapsto |00\rangle$, $|11\rangle \mapsto |10\rangle$, $|10\rangle \mapsto |11\rangle$. En d'autres mots, l'état

supposé $(|0\rangle + |1\rangle)/\sqrt{2}$ obtenu après une application de la porte Hadamard est en fait $|01\rangle$, et l'état supposé $(|0\rangle - |1\rangle)/\sqrt{2}$ est $|11\rangle$. L'appareil de mesure est lui aussi différent. Il mesure avec probabilité $1/2$ le premier qubit et en renvoie son résultat, sinon il fait de même avec le deuxième qubit. La mesure de $|00\rangle$ retourne toujours '0', la mesure $|11\rangle$ toujours '1'. L'appareil de mesure est donc fiable sur toute superposition de $|00\rangle$ et $|11\rangle$. En ce sens, l'appareil de mesure est fiable puisque fiable sur l'espace de dimension 2 généré par les encodages de $|0\rangle$ et $|1\rangle$. En revanche, en dehors de cet espace, l'appareil de mesure se comporte de manière inattendue. La mesure de $|01\rangle$ ou $|10\rangle$ retourne '0' ou '1' chacun avec probabilité $1/2$.

Ce système physique passe donc avec succès le test de [**DMMS00**, **DMMS07**] pour la porte Hadamard, alors qu'il ne la réalise clairement pas. Si ce système était utilisé dans la distribution quantique de clés secrètes de [BB84], le résultat serait désastreux au niveau de la sécurité cryptographique de ce protocole puisqu'un espion pourrait très simplement copier la clé distribuée. Il est en effet fondamental dans le protocole de [BB84] que les qubits échangés entre les participants soient piégés dans un espace de dimension 2.

2.4. Hypothèses de test. En respectant les règles du génie logiciel, nous devons commencer par préciser nos hypothèses de tests, *i.e.* les hypothèses que nous faisons sur le système testé :

H1: Le système physique sur lequel nous travaillons est constitué de plusieurs sous-systèmes identifiables.

H2: Deux sous-systèmes interagissent seulement si une porte prenant en entrée ces deux systèmes à la fois est appliquée.

H3: Chaque porte se comporte exactement de la même façon à chaque réalisation de la même expérience dans laquelle elle est utilisée.

H4: Tout calcul classique et tout contrôle classique sont exacts.

2.5. Simulation et équivalence. Dorénavant, au prix de ne connaître que partiellement l'espace physique du système testé, toute transformation sera supposée unitaire, tout état pur, et toute mesure de von Neumann. Dans ce cadre, la notion de test d'un dispositif quantique peut sembler bien vague. En effet, comment exprimer qu'une porte quantique est correcte si l'espace sur laquelle elle agit est inconnu ? Il nous faut donc préciser un peu plus notre cadre en introduisant les notions de simulation et d'équivalence.

Alors que la notion de simulation repose sur les statistiques fournies par les résultats d'une série d'expériences, l'équivalence est un concept mathématique portant sur la modélisation d'un système physique.

Etant donnée une famille fixée de mesures de von Neumann, servant de système de référence, un état *simule* un autre, si tous les deux produisent les mêmes statistiques de sortie pour toute la famille de mesures. Plus précisément, nous avons une famille de projecteurs $(P^w)_{w\in\mathcal{W}}$ (les mesures) agissant sur un espace *physique* H et un état $|\psi\rangle \in H$, supposés réaliser des projections fixées $|w\rangle\langle w|$ sur un espace *logique* (intuitivement l'espace de calcul) $H_c = \mathbb{C}^N$, pour un entier N fixé, et un état $|\phi\rangle \in H_c$. Moralement $N = 2^n$, où n est le nombre de qubits de l'espace logique.

Définition 5.2. *Un état quantique* $|\psi\rangle \in H$ simule *un état* $|\phi\rangle \in H_c$ *si pour tout* $w \in \mathcal{W}$ *:*

$$\|P^w|\psi\rangle\|^2 = |\langle w|\phi\rangle|^2 .$$

La notion de simulation peut aussi être transcrite sur l'espace physique H. Etant donnée une famille d'états $(|\psi_i\rangle)_{0\le i\le N-1}$ de H telle que chaque $|\psi_i\rangle$ simule l'état $|i\rangle$ de la base de calcul, alors on dira que H *simule* $\mathcal{H}_c$.

La simulation s'étend ensuite aux transformations unitaires.

Définition 5.3. *Supposons que* H *simule* H_c *:* $(|\psi_i\rangle)_i$ *simule* $(|i\rangle)_i$*. Une transformation unitaire* $G \in \mathcal{U}(H)$ simule *la transformation unitaire* $T \in \mathcal{U}(H_c)$ *si* $G|\psi_i\rangle$ *simule* $T|i\rangle$ *pour chacun des* $i = 0, 2, \ldots, N-1$.

Tester un circuit comme une seule opération unitaire n'est pas envisageable. En effet, cela demanderait un nombre de tests de l'ordre de N, et donc exponentiel en le nombre de qubits utilisés.

Nous voudrions plutôt tester individuellement chaque porte (qui agit sur un nombre constant de qubits, disons 3) qui constitue le circuit. Cependant ceci n'est pas possible directement car la notion de simulation ne se compose pas. Pour cette raison, nous avons besoin du concept d'équivalence.

La notion d'équivalence est implicitement présente dans les travaux de Mayers et Yao [MY98], mais pas explicitement formulée. Il s'agit d'une notion mathématique exprimant la possibilité de transférer un état physique vers un espace logique. Les espaces physique H et logique H_c étant a priori de dimensions différentes, l'équivalence ne peut porter qu'entre H_c et un sous-espace S de H. Vouloir envoyer S et H par une transformation unitaire définit une notion d'équivalence trop forte. En particulier car cette notion ne se marie pas bien avec le produit tensoriel et la composition de portes dans notre contexte. La raison étant assez subtile, nous ne l'aborderons pas ici.

Nous introduisons donc une notion un peu plus délicate faisant intervenir le produit tensoriel entre H et H_c, noté $\bar{H} = H_c \otimes H$. Dans cet espace, H identifié au sous-espace $|0\rangle \otimes H$ de $\bar{H}$.

Nous sommes maintenant prêts à définir la notion d'équivalence entre un sous-espace S de l'espace physique H et l'espace logique H_c. Pour ne pas être triviale, cette définition est définie relativement à une famille de projecteurs servant encore une fois de système de référence.

Pour deux transformations linéaires X, Y la notation $X =_S Y$ signifie que les transformations X, Y coïncident sur le sous-espace S.

Définition 5.4. *Soit $U \in \mathcal{U}(\bar{H})$. Un sous-espace S de H est* U-équivalent *à H_c, si pour tout $w \in \mathcal{W}$, $P^w =_S U^\dagger(|w\rangle\langle w| \otimes \mathrm{Id}_H)U$.*

Cette définition revient à dire que le diagramme suivant est commutatif :

$$\begin{array}{rcl} S & \xrightarrow{P^w} & S \\ U \downarrow & & \uparrow U^\dagger \\ \bar{H} & \xrightarrow{|w\rangle\langle w| \otimes \mathrm{Id}_H} & \bar{H} \end{array} .$$

Intuitivement, la transformation unitaire U assure que la correspondance entre le système physique et le système logique est bien définie sur S, de sorte qu'une mesure du système physique dans un état de S revient à transporter cet état sur l'espace logique, y effectuer la mesure idéale, puis de le retransporter sur l'espace physique. Ce procédé peut rappeler sous certains aspects la définition d'un appareil de mesure.

Une conséquence de cette définition est que les projecteurs P^w satisfont $P^w(S) \subseteq S$. Si de plus les états $|w\rangle$ génèrent tout H_c alors S est nécessairement de même dimension que H_c.

Enfin, définissons la notion d'équivalence pour les états et les portes.

Définition 5.5. *Soit S un sous-espace de H. Un état $|\psi\rangle \in S$ est* U-equivalent *à $|\phi\rangle \in H_c$ sur S si*

1. *S est U-équivalent à H_c,*
2. *$|\psi\rangle = U^\dagger(|\phi\rangle \otimes |\chi\rangle)$, pour un état $|\chi\rangle \in H$.*

Définition 5.6. *Soit S un sous-espace de H. Une transformation unitaire $G \in \mathcal{U}(H)$ est* (U, V)-équivalente *à $T \in \mathcal{U}(H_c)$ sur S si*

1. *S est U-équivalent à H_c,*
2. *$S' = G(S)$ est V-équivalent à H_c,*
3. *$G =_S V^\dagger(T \otimes W)U$, pour une transformation $W \in \mathcal{U}(H)$.*

Cette définition revient à dire que le diagramme suivant est commutatif :

$$\begin{array}{rcccccc} S & \xleftarrow{P^w} & S & \xrightarrow{G} & S' & \xrightarrow{P^w} & S' \\ U^\dagger \uparrow & & \downarrow U & & V^\dagger \uparrow \downarrow V & & \uparrow V^\dagger \\ \bar{H} & \xleftarrow{|w\rangle\langle w| \otimes \mathrm{Id}_H} & \bar{H} & \xrightarrow{T \otimes W} & \bar{H} & \xrightarrow{|w\rangle\langle w| \otimes \mathrm{Id}_H} & \bar{H} \end{array} .$$

L'état $|\chi\rangle$ de la définition de l'équivalence d'états est communément appelé *garbage* en anglais, car au moment où la transformation T, ou la projection $|w\rangle\langle w|$, est appliquée, cet état n'est pas du tout pris en compte. Cependant, l'indétermination de cet état justifie notre notion d'équivalence en tempérant ce qu'on peut espérer d'un système testé. En effet, la définition de l'équivalence de transformations unitaires laisse la possibilité à une porte de ne pas être l'identité sur l'espace garbage, à condition que cette transformation soit décorrélée de celle sur l'espace logique.

Concluons en énonçant une propriété attendue de l'équivalence : cette dernière implique la simulation.

LEMME 5.1. *Soit S un sous-espace de H. Supposons que $(|w\rangle)_{w\in\mathcal{W}}$ génèrent tout H_c. Soit $|\psi_i\rangle$ un vecteur unitaire de $P^i(S)$, pour $0 \leq i \leq N-1$. Si $G \in \mathcal{U}(H)$ est équivalent à $T \in \mathcal{U}(H_c)$ sur S, alors*

(1) *H simule H_c : $(|\psi_i\rangle)_i$ simule $(|i\rangle)_i$;*

(2) *G simule T.*

Remarquons que toutes ces notions sont exactes et ne semblent pas tolérer la moindre imperfection. En fait, comme nous le verrons dans le Théorème 5.6, tous nos résultats d'équivalence et de simulation seront à la fois *tolérants* et *robustes*. Notre testeur tolérera non seulement des dispositifs à faible taux d'erreur, étant donné un certain seuil donné, tout en rejetant ceux trop erronés, étant donné un autre seuil dépendant du premier de manière polynomiale (exposant 8).

2.6. Test d'une porte. Nous considérerons un *test* comme un ensemble de conditions de simulations, chacune d'elles pouvant être expérimentalement vérifiée à un seuil de précision donné, par la répétition d'une expérience.

Nous allons montrer comment construire des tests efficaces pour des circuits quantiques en étudiant des tests élémentaires portant uniquement sur des sources d'entrées et des portes, et en prouvant comment ces tests se composent pour caractériser le circuit complet.

Commençons par notre brique de base pour le test de sources. Il s'agit ni plus ni moins du résultat de Mayers et Yao [MY98] reformulé à l'aide des notions de simulation et d'équivalence.

A partir de maintenant nous posons $\mathcal{A}_0 = \{0, \frac{\pi}{8}, \frac{\pi}{4}\}$, $\mathcal{A}_1 = \{a + \frac{\pi}{2} : a \in \mathcal{A}_0\}$, et $\mathcal{A} = \mathcal{A}_0 \cup \mathcal{A}_1$. Nous fixons des ensembles de mesures orthogonales $(P_A^a, P_A^{a+\pi/2})_{a\in\mathcal{A}_0}$ et $(P_B^b, P_B^{b+\pi/2})_{b\in\mathcal{A}_0}$ sur respectivement deux espaces physiques A et B. Nous supposons donc que $P_A^a + P_A^{a+\pi/2} = \mathrm{Id}_A$ et $P_B^a + P_B^{a+\pi/2} = \mathrm{Id}_B$, pour tout $a \in \mathcal{A}_0$.

Enfin, pour $\alpha \in \mathbb{R}$, soit $|\alpha\rangle = \cos\alpha|0\rangle + \sin\alpha|1\rangle$. Donc en particulier, $|\frac{\pi}{2}\rangle = |1\rangle$. Notons $|\phi^+\rangle$ l'état EPR $\frac{1}{\sqrt{2}}(|0\rangle\otimes|0\rangle + |1\rangle\otimes|1\rangle)$, et $|\Phi_n^+\rangle$ le produit tensoriel de n états EPR : $|\Phi_n^+\rangle = \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|x\rangle\otimes|x\rangle$.

THÉORÈME 5.3. *Soient $H = A\otimes B$ et $|\psi\rangle \in H$ qui simule $|\phi^+\rangle$. Alors il existe deux transformations unitaires $U_{\bar{A}} \in \mathcal{U}(\bar{A})$ et $U_{\bar{B}} \in \mathcal{U}(\bar{B})$ telles que $|\psi\rangle$ est $(U_{\bar{A}} \otimes U_{\bar{B}})$-équivalent à $|\phi^+\rangle$ sur $S = \mathsf{Span}\{P_A^a \otimes P_B^b|\psi\rangle : a, b \in \mathcal{A}\}$. De plus, la dimension de S est 4.*

L'hypothèse de ce théorème est une condition de simulation qui exprime donc naturellement la séquence de tests suivante, dite *test EPR* :

$$\|P_A^a \otimes P_B^b|\psi\rangle\|^2 = \left|\langle ab|\phi^+\rangle\right|^2, \quad \text{pour tout } a, b \in \mathcal{A}.$$

Avant de décrire le test de portes, détaillons les caractéristiques que nous retrouverons tout au long de nos tests. Tout test requiert deux tests EPR qui assurent que les états d'entrées et de sorties sont cohérents avec les appareils de mesures avant et après la porte. Il s'agit des *tests de conspiration*. Ensuite, les statistiques de la porte sont testées sur les états EPR, leurs corrélations constituent les *tests de tomographie*.

Voici maintenant pour illustrer le type de résultats obtenus, le test d'une porte $T \in \mathcal{O}(2)$ sur 1 qubit à coefficients réels, où $\mathcal{O}(N)$ désigne la restriction de $\mathcal{U}(N)$ aux matrices à coefficients réels. Ce test utilise le fait qu'une telle porte T appliquée sur le premier qubit de l'état EPR $|\phi^+\rangle$ peut être défaite en appliquant la même porte T sur le deuxième qubit. Plus précisément, ce test vérifie non pas une porte mais une paire de portes (G_A, G_B), où G_A est censé réaliser T sur la partie A, et G_B réaliser T sur B.

THÉORÈME 5.4. *Soit $T \in \mathcal{O}(2)$. Soient $H = A \otimes B$, $G_A \in \mathcal{U}(A)$, et $G_B \in \mathcal{U}(B)$. Soit $|\psi\rangle \in H$ tel que $|\psi\rangle$ et $G_AG_B|\psi\rangle$ simulent tous deux $|\phi^+\rangle$, et tel que $G_A|\psi\rangle$ simule $(T \otimes \mathrm{Id}_2)|\phi^+\rangle$. Alors il existe $U_{\bar{A}} \in \mathcal{U}(\bar{A})$ et $U_{\bar{B}} \in \mathcal{U}(\bar{B})$ telles que G_A est $(U_{\bar{A}} \otimes U_{\bar{B}})$-équivalent à T sur $S = \mathsf{Span}\{P_A^a \otimes P_B^b|\psi\rangle : a, b \in \mathcal{A}\}$.*

Revenons sur la restriction aux portes à coefficients réels. Tout d'abord cette restriction ne concerne que la spécification, *i.e.* sur la porte idéale censée être réalisée. Aucune hypothèse n'est faite sur la réalisation physique de cette porte par la transformation G qui peut, elle, être *a priori* à coefficients

complexes. Le problème vient du fait que n'importe quelle porte complexe en dimension d peut être simulée [RG02] par une porte réelle et des mesures appropriées en dimension $2d$. D'un point de vue positif, cela signifie que notre restriction n'est pas une limitation, puisque n'importe quel calcul quantique peut être réalisé avec des portes que nous pouvons tester. D'un point de vue négatif, il est physiquement impossible de distinguer une porte complexe en dimension par exemple 2, d'une porte réelle en dimension 4.

Ce test peut ensuite être généralisé à des portes sur plusieurs qubits, et comme il se compose au sens de l'équivalence, il permet de tester un circuit entier en vérifiant au fur et à mesure chacune de ses portes. Ce test est décrit dans la section suivante, puis nous terminerons en considérant le test d'un circuit sur une entrée fixée, ce qui correspond à l'exécution d'un calcul quantique.

2.7. Test d'un circuit. Nous énonçons maintenant le théorème principal qui relie le test des sources et portes à la simulation du circuit tout entier. Cet énoncé regroupe le fait que sous certaines conditions la simulation implique l'équivalence, que les énoncés d'équivalence se composent, et que l'équivalence implique la simulation.

Supposons que l'espace physique H se décompose explicitement en produit tensoriel de n paires de registres : $H = \bigotimes_{i=1}^n A^i \bigotimes B^i$. Pour chaque ensemble de registres $I \subseteq \{1, 2, \ldots, n\}$, soient H^I l'espace $\bigotimes_{i\in I} A^i \bigotimes_{i\in I} B^i$, et $|\Phi^+\rangle_I$ le produit tensoriel de $|I|$ états EPR sur $\bigotimes_{i\in I} A^i_c \bigotimes_{i\in I} B^i_c$.

THÉORÈME 5.5. *Soit $H = A\otimes B$, où $A = \bigotimes_i A^i$ et $B = \bigotimes_i B^i$. Soient $I^1, I^2, \ldots, I^t \subseteq \{1, 2, \ldots, n\}$. Soient $G^j_A \in \mathcal{U}(A^{I^j})$, $G^j_B \in \mathcal{U}(B^{I^j})$ et $T^j \in \mathcal{O}(A^{I^j}_c)$. Soit $|\Psi\rangle \in A \otimes B$. Soient les états $|\Psi^j\rangle, |\Psi'^j\rangle$ définis récursivement par $|\Psi'^j\rangle = (G^j_A \otimes \mathrm{Id}_B)|\Psi^{j-1}\rangle$ et $|\Psi^j\rangle = (G^j_A \otimes G^j_B)|\Psi^{j-1}\rangle$, où $|\Psi^0\rangle = |\Psi'^0\rangle = |\Psi\rangle$. Supposons que :*

(1) *$|\Psi\rangle$ simule $|\phi^+\rangle$.*
(2) *Pour $j = 1, \ldots, t$: $|\Psi^j\rangle$ simule $|\phi^+\rangle$.*
(3) *Pour $j = 1, \ldots, t$: $|\Psi'^j\rangle$ simule $T^j|\Phi^+\rangle_{I^j}$.*

Alors $G^t_A G^{t-1}_A \cdots G^1_A$ est équivalent à $T^t T^{t-1} \cdots T^1$ sur $S = \mathsf{Span}(P^a_A \otimes P^b_B|\Psi\rangle : a, b \in \mathcal{A}^n)$.

COROLLAIRE 5.1. *Soit $|\Psi\rangle \in H$ qui satisfait les hypothèses du Théorème 5.5 pour une décomposition de $G_A \in \mathcal{U}(A)$ et $T \in \mathcal{U}(A_c)$ en t portes agissant sur un nombre constant de qubits (par exemple ≤ 3). Alors, pour tout $x \in \{0,1\}^n$, l'état $\sqrt{2^n}\,\mathrm{tr}_B(P^x_B|\Psi\rangle)$ simule $|x\rangle_{A_c}$. De plus, G_A simule T, et le nombre de statistiques à vérifier est en $O(t)$.*

2.8. Test d'un circuit sur une entrée fixée. De manière surprenante, il est plus facile de tester un circuit dans son ensemble que sur une entrée donnée. En fait, l'utilisation des états EPR permet de tester simultanément le circuit sur toutes les entrées possibles. Il n'est pas question de préparer l'entrée demandée à cause des contraintes de test qui nous interdisent de préparer nous même un état. On pourrait donc demander que cet état soit préparé pour nous, mais comment vérifier qu'il est correct ? Le choix fiable le plus simple serait de sélectionner l'entrée en mesurant les états EPR du côté B, et de recommencer jusqu'à ce que le résultat fournisse l'entrée voulue.

Nous contournons ce dilemme en supposant avoir un contrôle classique de nos circuits. Plus précisément en pouvant décider quelle porte appliquer du côté A, étant donné le résultat d'une mesure du côté B.

Soit donc une entrée binaire x de n bits, et un circuit censé réaliser une transformation unitaire T. Le but est d'observer l'application du circuit sur l'entrée x tout en vérifiant la validité du résultat. La première étape consiste à mesurer le côté de B de n états supposés EPR. Le résultat y de la mesure est donc censé fournir l'état classique y du côté A. Ensuite, du côté A un circuit $T_{x,y}$ est construit censé d'abord changer les valeurs des bits de y en ceux de x, puis d'appliquer le circuit initial pour T. Alors, ce circuit est exécuté sur l'état y du côté A et une mesure est effectuée. La partie du test proprement dite consiste alors à vérifier *a posteriori* que le circuit modifié $T_{x,y}$ est correct.

La Figure 5.2 décrit la suite des expériences à réaliser pour tester un circuit formé des portes $G^3_A G^2_A G^1_A$ sur l'entrée $|00\rangle$. Le calcul est d'abord effectué (Experiment 1) une fois sur le circuit modifié, où les mesures intermédiaires du côté B ont fourni les réponses M_1, M_2. Puis, pour vérifier que la sortie du circuit est correcte, chacune des expériences suivantes (Experiment 2–8) est réalisée $\log(n/\gamma)/\varepsilon$ fois, où ε est la précision demandée et γ un paramètre de confiance.

Voici maintenant notre auto-testeur générique, paramétré par :

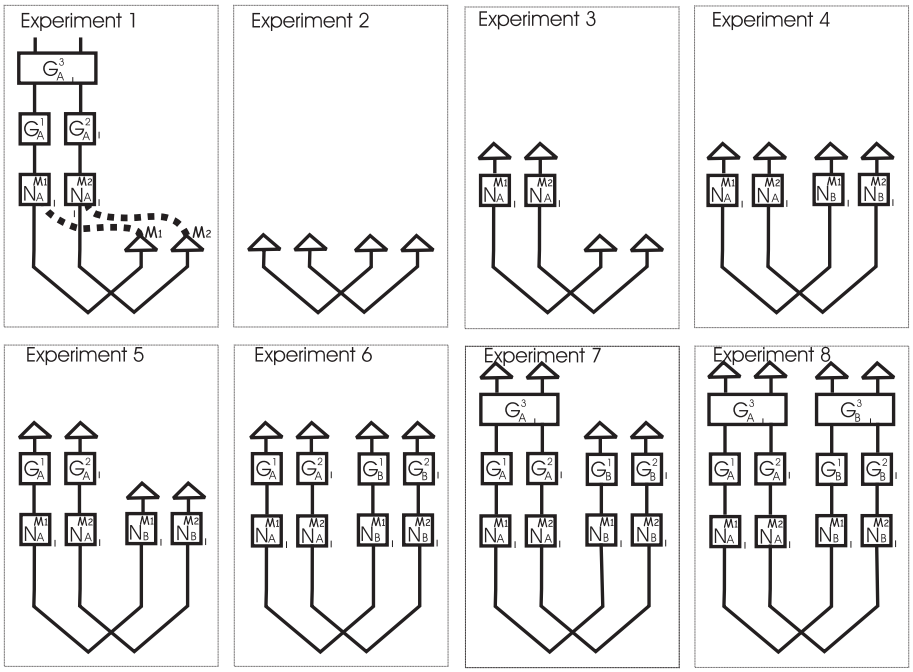

FIG. 5.2. Les expériences successives pour tester le circuit formé des portes $G_A^3 G_A^2 G_A^1$ sur l'entrée $|00\rangle$.

– un circuit $T \in \mathcal{U}(2^n)$, c'est-à-dire par une décomposition en portes agissant sur un nombre constant de qubits (par exemple ≤ 3) $T^t T^{t-1} \cdots T^1 = T$;
– une chaîne binaire $x \in \{0,1\}^n$;
– un paramètre de précision $\varepsilon > 0$, et un paramètre de confiance $\gamma > 0$.

L'entrée est une source d'états $|\Psi\rangle$ sur n paires de registres quantiques $A \otimes B = \bigotimes_i (A^i \otimes B^i)$, des portes G_A^j and G_B^j agissant sur les mêmes numéros de registres que T^j, des portes auxiliaires N_A^i agissant sur le i-ème registre A_i de A, des mesures orthogonales $(P_{A^i}^a, P_{A^i}^{a+\pi/2})_{a \in \mathcal{A}_0}$ et $(P_{B^i}^b, P_{B^i}^{b+\pi/2})_{b \in \mathcal{A}_0}$. Le but est de vérifier d'abord que $\sqrt{2^n}\,\mathrm{tr}_B(P_B^b|\Psi\rangle)$ simule $|b\rangle$, puis que le circuit $G_A = G_A^t G_A^{t-1} \cdots G_A^1$ simule T.

Les notations utilisées sont celles du Théorème 5.5.

Circuit Test $(T^1, T^2, \ldots, T^t \in \mathcal{U}(2^n), x \in \{0,1\}^n, \varepsilon > 0, \gamma > 0)$

(1) Préparer un état $|\Psi\rangle$ de n états EPR en n paires disjointes sur $A^1 \otimes B^1, \ldots, A^n \otimes B^n$
(2) Mesurer le côté de B de $|\Psi\rangle$ en utilisant les mesures $(P_B^b)_{b \in \{0,\pi/2\}^n}$, et soit y le résultat obtenu
(3) Soit $T_{x,y}$ le circuit qui envoie $|y\rangle$ sur $|x\rangle$ et puis qui applique T
(4) Préparer du côté A le circuit G_A réalisant $T_{x,y}$ à l'aide des t portes G_A^j et d'au plus n portes N_A^i. Soit $t' \leq t + n$ le nombre total de portes utilisées.
(5) Appliquer ce circuit du côté A et mesurer le résultat à l'aide de $(P_A^a)_{a \in \{0,\pi/2\}^n}$
(6) Approcher les statistiques suivantes en répétant $O(\frac{\log(n/\gamma)}{\varepsilon})$ fois les expériences correspondantes :
  (a) Mesurer $|\Psi\rangle$ à l'aide de $(P_{A^i}^{a^i} \otimes P_{B^i}^{b^i})_{a^i, b^i \in \mathcal{A}_0}$, pour chaque $i = 1, 2, \ldots, n$
  (b) Pour $j = 1, \ldots, t'$: Mesurer $|\Psi^j\rangle$ à l'aide de $(P_{A^i}^{a^i} \otimes P_{B^i}^{b^i})_{a^i, b^i \in \mathcal{A}}$, pour chaque $i \in I^j$
  (c) Pour $j = 1, \ldots, t'$: Mesurer $|\Psi'^j\rangle$ à l'aide de $(P_{A^{I^j}}^a \otimes P_{B^{I^j}}^b)_{a,b \in \mathcal{A}_0^{I^j}}$
(7) Accepter si et seulement toutes les statistiques sont correctes à une erreur additive près ε

THÉORÈME 5.6. *Soient* $T^1, T^2, \ldots, T^t \in \mathcal{U}(2^n), x \in \{0,1\}^n, \varepsilon > 0, \gamma > 0$.

Si ***Circuit Test***$(T^1, T^2, \ldots, T^t, x, \varepsilon, \gamma)$ *accepte, alors avec probabilité* $1 - O(\gamma)$*, la distribution de probabilité de sortie du circuit à l'étape (5) est à distance en variation*[1] $O((t+n)\varepsilon^{1/8})$ *de la distribution attendue, i.e. provenant de la mesure de* $T^t T^{t-1} \cdots T^1 |x\rangle$ *par* $(|a\rangle\langle a|)_{a \in \{0,\pi/2\}^n}$.

Inversement, si ***Circuit Test***$(T^1, T^2, \ldots, T^t, x, \varepsilon, \gamma)$ *rejette, alors avec probabilité* $1 - O(\gamma)$*, au moins un des éléments parmi l'état* $|\Psi\rangle$*, les portes* G_A^i, G_B^i *et* N_A^i *sont à distance*[2] *au moins* $\Omega(\varepsilon)$ *respectivement d'un des éléments* $|\Phi_n^+\rangle$*,* T^i, T^i *et* $\mathrm{NOT}_{A_c^i}$ *sur* $S = \mathsf{Span}(P_A^a \otimes P_B^b |\Psi\rangle : a, b \in \mathcal{A}^n)$.

De plus le nombre d'expériences réalisées par ***Circuit Test***$(T^1, T^2, \ldots, T^t, x, \varepsilon, \gamma)$ *est en* $O(\frac{tn}{\varepsilon} \log(n/\gamma))$.

[1] Il s'agit de la distance ℓ_1 sur les distributions de probabilité divisée par 2.

[2] Il s'agit de la distance associée à la norme d'opérateur pour la norme de H.

CHAPITRE 6

Perspectives

1. Vérification

1.1. Interaction avec les autres communautés de la vérification. Je souhaite continuer à poursuivre mes efforts d'interaction sur le thème de la vérification, en montrant que l'utilisation d'outils d'approximation probabiliste peut aider la vérification.

Il est parfois suffisant de savoir qu'un programme marche avec une certaine fiabilité, plutôt que d'attendre indéfiniment sa validation. Tout réside alors dans la quantification de cette fiabilité. Voici l'exemple de trois contextes où l'apport des techniques des méthodes probabilistes issues des algorithmes d'approximation et celles du property testing, semblent prometteurs.

Dans le cas du black-box testing [YL95], le programme est testé uniquement via une interaction depuis l'extérieur : à aucun moment son mécanisme n'est révélé. Ce contexte rappelle une des motivations initiales du self-testing, c'est pourquoi y confronter les méthodes probabilistes du self-testing semble intéressant. Ce contexte et ses variantes sont très étudiés en génie logiciel, où des approches probabilistes commencent à être considérées [GDGM01], mais il y manque une notion d'approximation qui permettrait de quantifier la qualité du test effectué.

Le black-box checking est une théorie relativement récente qui a réussi à combiner habilement le model checking et le black-box testing, par exemple suivant l'approche de [PVY02] mais sans notion d'approximation à notre connaissance. C'est pourquoi je voudrais reprendre cette problématique sous l'angle du property testing.

Enfin, dans le cadre des protocoles de communication, on peut voir le déroulement d'un protocole à plusieurs joueurs comme un arbre. Puisque nous avons montré que les propriétés régulières d'arbres pouvaient être testées, il reste à voir quel est concrètement cet apport pour des problèmes de sécurité et de sûreté par exemple. C'est un axe que je compte développer, en assimilant d'abord les travaux récents dans ce domaine comme [CC05].

1.2. Property testing et Streaming algorithms. Un autre courant concernant les masses de données (massive data set) concerne la vérification de flots (streams) au sens du property testing. Cette approche a été proposée dans [FKSV02], mais est encore sous-évaluée.

Les algorithmes de flots (streaming algorithms) traitent des données tellement gigantesques qu'ils ne peuvent les stocker en mémoire vive. Ils peuvent donc uniquement maintenir une mémoire logarithmique en la taille du flot, qui doit être mise à jour en temps logarithmique à chaque nouvelle entrée lue. Un des premiers domaines d'application fut sans doute l'étude de trafic internet où des résultats très importants ont été obtenus [AMS99]. Dans des domaines comme la vérification de séquences ADN, de longs fichier XML, ou plus généralement de bases de données gigantesques, les applications sont encore rares.

Alors qu'un testeur au sens du property testing ne peut regarder qu'une fraction constante de l'entrée, un streaming property tester peut lire toute l'entrée mais dans un ordre qui lui est imposé. Dans [FKSV02], il a été montré que ces deux notions pouvaient être radicalement différentes. Nous espérons donc trouver de nouveaux testeurs là où il n'en existait pas, ou encore améliorer les précédents. Des pistes semblent très prometteuses dans le contexte de la vérification de propriétés régulières d'arbres sur des arbres codés par des structures XML. Actuellement, dans le cas général, la donnée XML doit être transformée en arbre DOM pour pouvoir évaluer la propriété. Dans certains cas [SV02], il est effectivement possible de décider la propriété avec un algorithme de flot directement sur la donnée XML. J'espère pouvoir étendre très largement cette possibilité par l'utilisation des techniques probabilistes de [FKSV02].

2. Bornes inférieures probabilistes et quantiques

2.1. Méthodes par adversaires. Les méthodes par adversaires ont eu des impacts surprenants pour l'étude de la complexité des formules logiques. En effet, le carré d'un minorant de la complexité en requêtes d'une fonction booléenne f, fourni par une méthode d'adversaires, est aussi un minorant de la taille de f en tant que formule logique [LLS06].

Je pense qu'il existe encore d'autres passerelles non explorées entre le calcul quantique et d'autres domaines de la complexité. De nouvelles méthodes par adversaires viennent d'être établies, qui s'avèrent plus puissantes que les précédentes.

La première [AŠW06] a permis d'établir des compromis en complexité espace-temps, sur lesquels nous reviendrons plus tard.

La deuxième [HLŠ07] montre que dans la méthode par adversaires traditionnelle, les adversaires qui sont habituellement pondérés par des poids positifs, peuvent aussi l'être avec des poids négatifs. Plus formellement, la méthode spectrale décrite par le Théorème 4.3 reste non seulement valide lorsque les coefficients des matrices peuvent être négatifs, mais de plus les résultats obtenus sont parfois meilleurs.

Si le potentiel de ces méthodes est clair, il est encore mal cerné, sans doute parce qu'il est difficile de les appliquer. Je souhaite donc tenter de renouveler mes efforts de réécriture pour cette nouvelle méthode par adversaires, afin de mieux comprendre ces nouveaux outils. Une des pistes pourrait être d'utiliser la complexité de Kolmorov quantique [BDL01] plutôt que de se servir de la complexité de Kolmogorov classique.

Je voudrais tester ces nouvelles méthodes sur les deux problèmes décrits ci-dessous, et je souhaiterais aussi savoir si ces méthodes peuvent êtres transposées à d'autres domaines de la complexité, comme le calcul probabiliste. Contrairement aux méthodes par adversaires précédentes, tout prolongement de [HLŠ07] au calcul probabiliste semble ouvert.

2.2. Conjecture d'Aanderaa et Rosenberg. Rappelons brièvement cette conjecture détaillée à la Section 3.3. Ici, la complexité mesurée pour décider une propriété de graphe, est le nombre nécessaire d'arêtes qu'il faut connaître du graphe pour décider cette propriété. La version classique de cette conjecture dit que le nombre d'arêtes à évaluer d'un graphe à n sommets, pour décider une propriété monotone de graphe non constante est en $\Theta(n^2)$. La conjecture est prouvée dans le cas déterministe, mais toujours ouverte en probabiliste. Le meilleur résultat est $\Omega(n^{4/3}\log^{1/3} n)$ [CK01] qui provient d'un raffinement de [Haj91] qui donnait $\Omega(n^{4/3})$.

Cette conjecture devient pour le calcul quantique $\Omega(n)$, puisque toutes les situations peuvent se produire entre n et n^2. Toutefois les seuls résultats obtenus sont les racines carrées des résultats probabilistes ci-dessus. Ils proviennent de leurs adaptations directes [Yao03] via la méthode d'adversaires d'Ambainis.

Toutefois, le dernier résultat probabiliste [OSSS05] semble dévier des précédents dans le sens qu'il semble plus difficilement transposable. Ce résultat n'améliore pas systématiquement les précédents, mais les simplifie et fournit une nouvelle vision de la conjecture. Je souhaite l'étendre au quantique, ou bien comprendre pourquoi il ne s'y applique pas. J'espère ainsi progresser dans cette conjecture non seulement pour le calcul quantique, mais par ricochet aussi pour le calcul probabiliste. La preuve de résultats classiques par le biais d'un résultat quantique est maintenant un fait de plus en plus courant, ce qui légitime d'ailleurs l'étude du calcul quantique, même si l'ordinateur quantique venait à ne jamais exister.

2.3. Complexité espace-temps. Dans la description de nos algorithmes quantiques pour Triangle Finding à la Section 3.3 , le lien entre l'espace utilisé et la complexité en requêtes a été brièvement abordé. Actuellement, de tels compromis ont pu être quantifiés seulement pour des problèmes calculatoires à plusieurs sorties comme le tri d'un tableau. Concernant ce problème, les travaux de [KSW04] donnent un compromis de $T^2S = \Omega(n^3)$ pour trier une liste de taille N, où T représente le nombre de requêtes et S l'espace utilisé en nombre de bits, alors que classiquement le compromis est de $TS = \Omega(n^2)$. Notons que les deux compromis sont à peu près optimaux. La méthode consiste à utiliser les différentes sorties du problème et à considérer les sous-problèmes de décision sous-jacents.

Cette approche est donc tout simplement inapplicable pour les problèmes de décision. Je voudrais donc développer de nouvelles techniques pour établir de tels compromis pour des problèmes

tels que Element Distinctness. Je compte m'attaquer à l'extension quantique des résultats classiques connus [BFH+87, Yao94] en utilisant une des générations récentes de la méthode par adversaires [AŠW06, HLŠ07].

3. Algorithmique quantique

3.1. Hidden Subgroup. D'un point de vue purement algorithmique, l'élément essentiel qui a permis de construire des algorithmes quantiques polynomiaux pour les instances abéliennes de Hidden Subgroup, est une procédure quantique permettant de réaliser la transformée de Fourier sur un groupe abélien en temps polylogarithmique par rapport au cardinal du groupe considéré. De telles procédures efficaces ont été trouvées pour toute une série de groupes non abéliens intéressants dont le groupe symétrique [Bea97]. Cependant l'utilité de ces transformées est méconnue, et très peu de tentatives utilisant ces transformations ont actuellement abouti [MRRS04].

Nos résultats concernant Hidden Subgroup utilisent la transformée de Fourier uniquement sur des sous-groupes abéliens. Je vais donc tenter de continuer à exploiter la transformée de Fourier sur les groupes abéliens pour résoudre des instances non abéliennes du problème du sous-groupe caché. L'une des perspectives de ces travaux serait de se rapprocher du cas des groupes symétriques qui incluent le problème de l'isomorphisme de graphes.

Nous avons montré qu'il était possible de résoudre Hidden Subgroup en temps quantique pseudo-polynomial pour les groupes résolubles d'exposants constants (Théorème 3.10), ainsi qu'en temps quantique sous-exponentiel pour les groupes résolubles quelconques (Théorème 3.11).

Deux voies sont alors possibles. L'une consiste à transformer ces algorithmes en algorithmes polynomiaux, ce qui exige de résoudre le problème sur le groupe dihédral $\mathbb{Z}_n \rtimes \mathbb{Z}_2$ en temps quantique polynomial. Rappelons que le meilleur algorithme quantique connu [Kup05] pour ce groupe est sous-exponentiel.

L'autre reflétant plus la tendance actuelle, consiste à classifier les groupes pour lesquels un algorithme pseudo-polynomial ou sous-exponentiel existe. J'ai de bons espoirs que des progrès significatifs dans les prochaines années soient obtenus dans ce sens pour le problème de l'isomorphisme de graphes.

3.2. Chaînes de Markov. Notre méthode de recherche basée sur les chaines de Markov (Théorème 3.15) ne permet pas de retrouver un résultat obtenu par Szegedy [Sze04], à savoir que toute marche quantique détecte la présence d'un élément marqué quadratiquement plus rapidement que le temps moyen pour atteindre un élément marqué (average hitting time) pour la marche aléatoire correspondante. Ce résultat permet de trouver un élément sur une grille en dimension 2 de taille $\sqrt{n} \times \sqrt{n}$ en $\sqrt{n} \log n$ déplacements, alors que le seul minorant connu sur le nombre minimal de déplacements est $\sqrt{n}$.

Il semblerait en fait que notre technique permette d' "oublier" certaines des valeurs propres de la chaîne de Markov utilisée, permettant d'en augmenter son écart propre, qui est un des paramètres cruciaux du Théorème 3.15. Les mathématiques mises en jeu étant très fines, nous n'avons pas pu encore aboutir. J'ai toutefois bon espoir d'y arriver et de retrouver non seulement le résultat de Szegedy pour la grille, mais de l'améliorer afin de demontrer qu'il est possible de résoudre ce problème en seulement $\sqrt{n}$ déplacements.

Un autre champ d'applications des marches quantiques a été récemment ouvert par Richter [Ric07]. Il montre que sous certaines conditions il est possible de mélanger quadratiquement plus rapidement avec une marche quantique qu'avec une marche aléatoire. Si ce résultat venait à être généralisé pour les chaînes de Markov habituellement utilisées en algorithmique, alors les algorithmes tels que ceux de Schöning [Sch02] pour 3-SAT où de Jerrum, Sinclair et Vigoda [JSV04] pour l'approximation du permanent, obtiendraient systématiquement des gains polynomiaux en calcul quantique. Pour ces deux derniers problèmes, seules les approches triviales utilisant l'algorithme de Grover sont actuellement connues en calcul quantique. Je pense que cette nouvelle voie est donc à suivre de très près. L'important ici étant de décrire une nouvelle méthodologie plutôt que de résoudre un problème particulier. L'apport serait un gain systématique à tout algorithme utilisant un échantillonage uniforme à base de marches aléatoires.

4. Cryptographie quantique

Les deux résultats majeurs en cryptographie quantique sont radicalement opposés. Le premier est l'existence d'un protocole quantique de distribution de clé secrète [BB84], qui a été prouvé inconditionnellement sûr assez récemment contre tout type d'attaques quantiques [SP00], y compris en présence d'appareils imparfaits [GLLP04]. Ce résultat est à contraster avec les résultats de cryptographie classique dont la sécurité repose toujours sur la difficulté présupposée, et donc non prouvée, d'un problème combinatoire. Ce protocole est de plus réalisable sur des distances de l'ordre de 100 km en utilisant de la fibre optique. Des expériences sont en cours pour passer à une transmission dans l'air qui permettrait de réaliser ce protocole par le biais de satellites. Il s'agit donc d'un protocole qui a tous les atouts de son côté.

Inversement, l'autre résultat est la preuve de l'impossibilité de réaliser une primitive cryptographique importante, qui s'appelle la mise en gage de bit. Le but de cette primitive est d'envoyer un bit dans un coffre fort numérique, que seul l'envoyeur sait déchiffrer, tout en étant dans l'impossibilité d'en changer la valeur une fois le coffre envoyé à son destinataire. Il a été prouvé [May97, LC98] qu'un ordinateur quantique ne pouvait pas réaliser une telle primitive qui soit inconditionnellement sûre.

Pendant une longue période très peu d'autres résultats marquants sont venus enrichir cette discipline. Cependant, depuis que ce résultat d'impossibilité a été assimilé, de nouvelles découvertes récentes nous laissent espérer des perspectives prometteuses dans cet axe de recherche.

4.1. Tirage à pile ou face à distance. La primitive de tirage à pile ou face à distance introduite par Blum [Blu81] consiste pour deux participants distants l'un de l'autre à se mettre d'accord sur un bit aléatoire. Ils disposent de sources aléatoires privées, mais pas de source aléatoire partagée.

Tout comme la mise en gage de bit, le tirage à pile ou face à distance ne peut être réalisé parfaitement et inconditionnellement en quantique [LC98], tout comme en classique.

Cependant il est possible d'étudier le biais $\varepsilon > 0$ qu'un protocole pourrait garantir. Un protocole est à biais ε, si chacune des valeurs 0 et 1 ne peut se produire qu'avec probabilité au plus $1/2 + \varepsilon$, et ce même si un des participants essaye de tricher. Si les protocoles classiques ne peuvent garantir aucun biais, la situation en quantique est meilleure. En effet, il existe un protocole [Amb04] qui garantit un biais au plus $1/4$. Un résultat d'impossibilité existe aussi [Kit02, GW07] : aucun biais ne peut être garanti en dessous de $(\sqrt{2}-1)/2 \approx 0.692$ par un protocole quantique.

Une version faible du tirage a aussi été étudiée. Elle est suffisante pour résoudre des conflits entre deux joueurs. Intuitivement, un des joueurs parie 0 et l'autre 1. Les joueurs exécutent un tirage à pile ou face à distance. Le gagnant est celui qui a parié sur la bonne valeur du tirage. Ainsi, si un des joueurs veut tricher, l'autre joueur sait de quel côté le premier veut biaiser la probabilité du tirage.

Pour cette version, la seule contrainte actuellement connue est que, pour atteindre un biais ε, il faut au moins $\log\log(1/\varepsilon)$ aller-retour de communication entre les deux joueurs. La version faible est en fait moins comprise. La situation est restée en suspens avec un protocole [SR02] garantissant un biais à $(\sqrt{2}-1)/2 \approx 0.207$, et donc coïncidant avec le biais de la version forte. Cependant, Mochon [Moc04] a franchi cette barrière en construisant un protocole à biais garanti égal à 0.192, puis rapidement après en montrant qu'il pouvait atteindre [Moc05] arbitrairement près le biais $1/6 \approx 0.167$.

Nous avons déjà lancé une recherche sur ces deux versions de tirage à distance avec des collègues et à l'occasion de l'encadrement d'un stage.

4.2. Mise en gage avec deux prouveurs. Il semblerait [CSST] que la mise en gage quantique soit possible en présence de deux prouveurs. Dans ce contexte, deux personnes mettent en gage le même bit, mais lors de la révélation du bit, ces deux personnes ne peuvent plus communiquer entre elles.

Si l'espoir persiste ici, la preuve n'est actuellement pas complète. Néanmoins c'est une ouverture que nous comptons approfondir.

4.3. Variables continues. Il s'agit ici d'un projet que je viens de démarrer avec un étudiant en stage. Les variables continues sont un support d'information quantique parfois plus adapté à la mise en œuvre pratique. Il y a peu de temps encore, seuls des systèmes quantiques vivant dans des espaces de dimension finie étaient étudiés pour la cryptographie. Lorsque le système est de dimension infinie non dénombrable, on parle alors de système à variables continues. Ces variables peuvent représenter, par exemple, des positions ou des quantités de mouvement.

Récemment, Grosshans *et al* [GVW$^+$03] ont proposé un protocole de distribution quantique de clé à base de variables continues, dont la sécurité a aussi été prouvée. Il s'agit donc d'un grand pas ouvrant le champ d'application de la cryptographie quantique à de nouveaux supports physiques. Si la distance n'est actuellement que de l'ordre de 10 km, le débit est lui bien plus élevé, car les impulsions utilisées ne contiennent plus un photon mais un paquet de photons, et sont donc plus facilement réalisables à un haut débit.

Il n'y a pas de correspondances claires entre les systèmes cryptographiques quantiques discrets et ceux continus. C'est pourquoi il faut réétudier un à un les résultats discrets en continu. Je voudrais notamment vérifier que la mise en gage quantique est toujours impossible, et savoir si les biais du tirage à pile ou face restent inchangés.

Ce projet pourra se mener avec des collaborations qui ont existé dans le passé avec l'équipe de Grangier de l'Institut d'Optique, et avec qui je continue d'entretenir des liens.

Bibliographie

[Aar04] S. Aaronson. Lower bounds for local search by quantum arguments. In *Proceedings of 36th ACM Symposium on Theory of Computing*, 2004. To appear. Also in quant-ph/0307149.

[Aar05] S. Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society A*, 461(2063) :3473–3482, 2005.

[ADR82] A. Aspect, J. Dalibard, and G. Roger. Experimental test of Bell's inequalities using time-varying analyzers. *Physical Review Letters*, 49(25) :1804–1807, 1982.

[AF06] David Aldous and James A. Fill. Reversible markov chains and random walks on graphs [online]. 2006. Available from : `http://www.stat.berkeley.edu/users/aldous/RWG/book.html`. Monograph in preparation, August 2006 version.

[AFKS00] N. Alon, E. Fischer, M. Krivelevich, and M. Szegedy. Efficient testing of large graphs. *Combinatorica*, 20 :451–476, 2000.

[AG97] A. Apostolico and Z. Galil. *Pattern Matching Algorithms*, chapter 14 : Approximate Tree Pattern Matching. Oxford University Press, 1997.

[AGR82] A. Aspect, P. Grangier, and G. Roger. Experimental realization of Einstein-Podolsky-Rosen-Bohm *Gedankenexperiment* : a new violation of Bell's inequalities. *Physical Review Letters*, 49(2) :91–94, 1982.

[AK02] N. Alon and M. Krivelevich. Testing k-colorability. *SIAM Journal on Discrete Mathematics*, 15 :211–227, 2002.

[AKK99] S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. *Journal of Computer and System Sciences*, 58(1) :193–210, 1999.

[AKNS00] N. Alon, M. Krivelich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. *SIAM Journal on Computing*, 30(6), 2000.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problem. *Journal of the ACM*, 45 :501–555, 1998.

[Amb02] A. Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64 :750–767, 2002.

[Amb03a] A. Ambainis. Polynomial degree vs. quantum query complexity. In *Proceedings of 44th IEEE Symposium on Foundations of Computer Science*, pages 230–239, 2003.

[Amb03b] A. Ambainis. Quantum walk algorithm for element distinctness. Technical Report quant-ph/0311001, arXiv, 2003.

[Amb04] A. Ambainis. Quantum walk algorithm for element distinctness. In *Proceedings of 45th Symposium on Foundations of Computer Science*, pages 22–31, 2004.

[AMS99] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1) :137–147, 1999.

[AS98] S. Arora and S. Safra. Probabilistic checking of proofs : A new characterization of NP. *Journal of the ACM*, 45(1) :70–122, 1998.

[AŠW06] A. Ambainis, R. Špalek, and R. de Wolf. A new quantum lower bound method : with applications to direct product theorems and time-space tradeoffs. In *Proceedings of the 38th ACM Symposium on Theory of computing*, pages 618–633, 2006.

[BB84] C. H. Bennett and G. Brassard. Quantum cryptography : Public key distribution and coin tossing. In *Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179, 1984.

[BB93] R. Beals and L. Babai. Las Vegas algorithms for matrix groups. In *Proceedings of 34th IEEE Symposium on Foundations of Computer Science*, pages 427–436, 1993.

[BBC+01] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4) :778–797, 2001.

[BBF+01] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and P. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools*. Springer, 2001.

[BCC+95] A. Barenco, C.Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Physical Review A*, 52(5) :3457–3467, 1995.

[BCWZ99] H. Buhrman, R. Cleved, R. de Wolf, and C. Zalka. Bounds for small-error and zero-error quantum algorithms. In *Proceedings of 40th Symposium on Foundations of Computer Science*, pages 358–368, 1999.

[**BDH$^+$01**] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf. Quantum algorithms for element distinctness. In *Proceedings of 15th IEEE Conference on Computational Complexity*, pages 131–137, 2001. Journal version in [**BDH$^+$05**].

[**BDH$^+$05**] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf. Quantum algorithms for element distinctness. *SIAM Journal on Computing*, 34(6) :1324–1330, 2005. Journal version of [**BDH$^+$01**]. Available from : `http://www.lri.fr/~magniez/PAPIERS/bdhhmsw-sicomp05.pdf`.

[BDL01] A. Berthiaume, W. van Dam, and S. Laplante. Quantum kolmogorov complexity. *Journal of Computer and System Sciences*, 63(2) :201–221, 2001.

[Bea97] R. Beals. Quantum computation of Fourier transforms over symmetric groups. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 48–53, 1997.

[BEK$^+$03] T. Batu, F. Ergün, J. Kilian, A. Magen, S. Raskhodnikova, R. Rubinfeld, and R. Sami. A sublinear algorithm for weakly approximating edit distance. In *Proceedings of the 35th ACM Symposium on Theory of Computing*, pages 316–324, 2003.

[Bel64] J. Bell. On the Einstein-Podolsky-Rosen paradox. *Physics*, 1 :195–200, 1964.

[BFH$^+$87] A. Borodin, F. Fich, F. Meyer Auf Der Heide, E. Upfal, and A. Wigderson. A time-space tradeoff for element distinctness. *SIAM Journal on Computing*, 16(1) :97–99, 1987.

[BFNR03] H. Buhrman, L. Fortnow, I. Newman, and H. Röhrig. Quantum property testing. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete algorithms*, pages 480–488, 2003.

[BHT97] G. Brassard, P. Høyer, and A. Tapp. Quantum algorithm for the collision problem. *ACM SIGACT News (Cryptology Column)*, 28 :14–19, 1997.

[BK95] M. Blum and S. Kannan. Designing programs that check their work. *Journal of the ACM*, 42(1) :269–291, 1995.

[BLR93] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3) :549–595, 1993.

[Blu81] M. Blum. Coin flipping by telephone. In *Proceedings of 1st International Crytology Conference*, pages 11–15, 1981.

[Bro97] A. Broder. On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences*, pages 21–30, 1997.

[BŠ06] H. Buhrman and R. Špalek. Quantum verification of matrix products. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithm*, pages 880–889, 2006.

[BSS03] H. Barnum, M. Saks, and M. Szegedy. Quantum decision trees and semidefinite programming. In *Proceedings of the 18th IEEE Conference on Computational Complexity*, pages 179–193, 2003.

[BV97] E. Bernstein and U. Vazirani. Quantum complexity theory. *SIAM J. Comp.*, 26(5) :1411–1473, 1997.

[CC05] H. Comon and V ? Cortier. Tree automata with one memory set constraints and cryptographic protocols. *Theoretical Computer Science*, 331(1) :143–214, 2005.

[CDG$^+$97] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on : `http://www.grappa.univ-lille3.fr/tata`, 1997. release October, 1rst 2002.

[CK01] A. Chakrabarti and S. Khot. Improved lower bounds on the randomized complexity of graph properties. In *Proceedings of 28th International Colloquium on Automata, Languages and Programming*, pages 285–296, 2001.

[CK02] H. Chockler and O. Kupferman. ω-regular languages are testable with a constant number of queries. In *Proceedings of the 6th Workshop on Randomization and Approximation Techniques in Computer Science*, pages 26–38, 2002. LNCS volume 2483.

[CM02] G. Cormode and S. Muthukrishnan. The string edit distance matching problem with moves. In *Proceedings of the 13th ACM-SIAM Symposium on Discrete algorithms*, pages 667–676, 2002.

[CSST] C. Crépeau, L. Salvail, J.-R. Simard, and A. Tapp. Classical and quantum strategies for two-prover bit commitments. Talk given for QIP'06 at IHP, Paris. Available from : `http://crypto.cs.mcgill.ca/~crepeau/QIP06-NB.pdf`.

[Deu85] D. Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. In *Proc. Roy. Soc. London*, volume 400 of *A*, pages 97–117, 1985.

[Deu89] D. Deutsch. Quantum computational networks. In *Proc. Roy. Soc. London*, volume 425 of *A*, pages 73–90, 1989.

[DHHM04] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. In *Proceedings of 31st International Colloquium on Automata, Languages and Programming*, pages 481–493, 2004.

[**DMMS00**] W. van Dam, F. Magniez, M. Mosca, and M. Santha. Self-testing of universal and fault-tolerant sets of quantum gates. In *Proceedings of 32nd ACM Symposium on Theory of Computing*, pages 688–696, 2000. Journal version in [**DMMS07**].

[**DMMS07**] W. van Dam, F. Magniez, M. Mosca, and M. Santha. Self-testing of universal and fault-tolerant sets of quantum gates. *SIAM Journal on Computing*, 2007. Journal version of [**DMMS00**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/dmms-sicomp07.pdf`.

[EH00] M. Ettinger and P. Høyer. On quantum algorithms for noncommutative hidden subgroups. *Advances in Applied Mathematics*, 25(3) :239–251, 2000.

[EKS00] F. Ergün, S. Kumar, and D. Sivakumar. Self-testing without the generator bottleneck. *SIAM Journal on Computing*, 29(5) :1630–1651, 2000.

[EPR35] A. Einstein, B. Podolsky, and N. Rosen. Can quantum-mechanical description of physical reality be considered complete ? *Physical Reviews*, 47(10) :777–780, 1935.

[Erg95] F. Ergün. Testing multivariate linear functions : Overcoming the generator bottleneck. In *Proceding of the 27th ACM Symposium on Theory of Computing*, pages 407–416, 1995.

[Fer96] W. Fernandez de la Vega. MAX-CUT has a randomized approximation scheme in dense graphs. *Random Structures Algorithms*, 8 :187–198, 1996.

[FFT] FFTW is a free collection of fast C routines for computing the Discrete Fourier Transform in one or more dimensions. It was developed at MIT by M. Frigo and S. Johnson. Available from : `htpp://www.fftw.org`.

[FGL+96] Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43 :268–292, 1996.

[**FIM+03**] K. Friedl, G. Ivanyos, F. Magniez, M. Santha, and P. Sen. Hidden translation and orbit coset in quantum computing. In *Proceedings of 35th ACM Symposium on Theory of Computing*, pages 1–9, 2003. Available from : `http://www.lri.fr/~magniez/PAPIERS/fimss-stoc03.pdf`.

[FK98] Uriel Feige and Joe Kilian. Zero knowledge and the chromatic number. *Journal of Computer and System Sciences*, 57(2) :187–199, 1998.

[FKSV02] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. Testing and spot-checking of data streams. *Algorithmica*, 34(1) :67–80, 2002.

[FM06] E. Fischer and A. Matsliah. Testing graph isomorphism. In *17th ACM-SIAM Symposium on Discrete Algorithms*, pages 299–308, 2006.

[**FMR06**] E. Fischer, F. Magniez, and M. de Rougemont. Approximate satisfiability and equivalence. In *Proceedings of 21st IEEE Symposium on Logic in Computer Science*, pages 421–430, 2006. Available from : `http://www.lri.fr/~magniez/PAPIERS/fmr-lics06.pdf`.

[**FMSS03**] K. Friedl, F. Magniez, M. Santha, and P. Sen. Quantum testers for hidden group properties. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science*, volume 2747 of *Lecture Notes in Computer Science*, pages 419–428. Springer, 2003. Available from : `http://www.lri.fr/~magniez/PAPIERS/fmss-mfcs03.pdf`.

[GDGM01] S. Gouraud, A. Denise, M.-C. Gaudel, and B. Marre. A new way of automating statistical testing methods. In *Proceedings of the 16th IEEE International Conference on Automated Software Engineering*, pages 5–12, 2001.

[GGR98] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4) :653–750, 1998.

[Gla89] S. Glasby. The composition and derived lengths of a soluble group. *Journal of Algebra*, 120 :406–413, 1989.

[GLLP04] D. Gottesman, H.-K. LO, N. Lutkenhaus, and J. Preskill. Security of quantum key distribution with imperfect devices. *Quantum Information and Computation*, 5 :325, 2004.

[Gro96] L. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 212–219, 1996.

[GSVV01] M. Grigni, L. Schulman, M. Vazirani, and U. Vazirani. Quantum mechanical algorithms for the nonabelian hidden subgroup problem. In *Proceedings of 33rd ACM Symposium on Theory of Computing*, pages 68–74, 2001.

[GVW+03] F. Grosshans, G. Van Assche, J. Wenger, R. Brouri, N. Cerf, and P. Grangier. Quantum key distribution using gaussian-modulated coherent states. *Nature*, 421 :238, 2003.

[GW07] G. Gutoski and J. Watrous. Toward a general theory of quantum games. In *Proceedings of 39th ACM Symposium on Theory of Computing*, 2007. To appear.

[Haj91] P. Hajnal. An $n^{4/3}$ lower bound on the randomized complexity of graph properties. *Combinatorica*, 11 :131–143, 1991.

[HH00] L. Hales and S. Hallgren. An improved quantum Fourier transform algorithm and applications. In *Proceedings of 41st IEEE Symposium on Foundations of Computer Science*, pages 515–525, 2000.

[HLŠ07] P. Høyer, T. Lee, and R. Špalek. Negative weights make adversaries stronger. In *Proceedings of 39th ACM Symposium on Theory of Computing*, 2007. To appear.

[HMRAR98] M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors. *Probabilistic methods for algorithmic discrete mathematics*, volume 16 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1998.

[HMT88] A. Hajnal, W. Maass, and G. Turán. On the communication complexity of graph properties. In *Proceedings of the 20th ACM Symposium on the Theory of Computing*, pages 186–191, 1988.

[HRT00] S. Hallgren, A. Russell, and A. Ta-Shma. Normal subgroup reconstruction and quantum computation using group representations. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 627–635, 2000.

[**IMS01**] G. Ivanyos, F. Magniez, and M. Santha. Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem. In *Proceedings of 13th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 263–270, 2001. Journal version in [**IMS03**].

[**IMS03**] G. Ivanyos, F. Magniez, and M. Santha. Efficient quantum algorithms for some instances of the non-abelian hidden subgroup problem. *International Journal of Foundations of Computer Science*, 14(5) :723–740, 2003. Journal version of [**IMS01**]. Available from : `http://www.lri.fr/~magniez/PAPIERS/ims-ijfcs03.pdf`.

[JSV04] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51(4) :671–697, 2004.

[Kit02] A. Kitaev. Quantum coin-flipping. Talk given for QIP'02 at MSRI, Berkeley, 2002. Available from : `http://www.msri.org/publications/ln/msri/2002/qip/kitaev/1/index.html`.

[**KLM06**] J. Kempe, S. Laplante, and F. Magniez. Comment calculer quantique. *La Recherche*, 398 :30–37, June 2006. Available from : `http://www.lri.fr/~magniez/PAPIERS/LaRecherche.pdf`.

[**KMS99**] M. Kiwi, F. Magniez, and M. Santha. Approximate testing with relative error. In *Proceedings of 31st ACM Symposium on Theory of Computing*, pages 51–60, 1999. Journal version in [**KMS03**].

[**KMS00**] M. Kiwi, F. Magniez, and M. Santha. Exact and approximate testing/correcting of algebraic functions : A survey. In *Proceedings of 1st Summer School on Theoretical Aspects of Computer Science*, volume 2292 of *Lecture Notes in Computer Science*, pages 30–83. Verlag, 2000. Also ECCC Report TR01-014. Available from : `http://www.lri.fr/~magniez/PAPIERS/kms-tehran00.ps.gz`.

[**KMS03**] M. Kiwi, F. Magniez, and M. Santha. Approximate testing with error relative to input size. *Journal of Computer and System Sciences*, 66(2) :371–392, 2003. Journal version of [**KMS99**]. Available from : `http://www.lri.fr/~magniez/PAPIERS/kms-jcss03.pdf`.

[KN97] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[KSV02] A. Kitaev, A. Shen, and M. Vyalyi. Classical and quantum computation. In *Graduate Studies in Mathematics*. American Mathematical Society, 2002.

[KSW04] H. Klauck, R. Spalek, and R. de Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. Technical Report quant-ph/0402123, arXiv, 2004.

[Kup05] G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM Journal on Computing*, 35(1) :170–188, 2005.

[KW04] I. Kerenidis and R. de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences*, 69(3) :395–420, 2004.

[LC98] H. K. Lo and H. F. Chau. Why quantum bit commitment and ideal quantum coin tossing are impossible. *Physica D*, 120 :177, 1998.

[**LLM$^+$02**] S. Laplante, R. Lassaigne, F. Magniez, S. Peyronnet, and M. de Rougemont. Probabilistic abstraction for model checking : An approach based on property testing. In *Proceedings of 17th IEEE Symposium on Logic in Computer Science*, pages 30–39, 2002. Journal version in [**LLM$^+$06**].

[**LLM$^+$06**] S. Laplante, R. Lassaigne, F. Magniez, S. Peyronnet, and M. de Rougemont. Probabilistic abstraction for model checking : An approach based on property testing. *ACM Transactions on Computational Logic*, 2006. Journal version of [**LLM$^+$02**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/llmpr-tocl06.pdf`.

[LLS06] S. Laplante, T. Lee, and M. Szegedy. The quantum adversary method and classical formula size lower bounds. *Computational Complexity*, 15(2) :163–196, 2006.

[**LM04**] S. Laplante and F. Magniez. Lower bounds for randomized and quantum query complexity using Kolmogorov arguments. In *Proceedings of 19th IEEE Conference on Computational Complexity*, pages 214–304, 2004. Journal version [**LM06**].

[**LM06**] S. Laplante and F. Magniez. Lower bounds for randomized and quantum query complexity using Kolmogorov arguments. *SIAM Journal on Computing*, 2006. Journal version of [**LM04**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/lm-sicomp06.pdf`.

[**Mag00a**] F. Magniez. *Auto-test pour les calculs approché et quantique*. PhD thesis, Université Paris-Sud, France, 2000. Library number 6076. Available from : `http://www.lri.fr/~magniez/PAPIERS/these.ps.gz`.

[**Mag00b**] F. Magniez. Multi-linearity self-testing with relative error. In *Proceedings of 17th Symposium on Theoretical Aspects of Computer Science*, volume 1770 of *Lecture Notes in Computer Science*, pages 302–313. Verlag, 2000. Journal version in [**Mag05**].

[**Mag05**] F. Magniez. Multi-linearity self-testing with relative error. *Theory of Computing Systems (TOCS)*, 38(5) :573–591, 2005. Journal version of [**Mag00b**]. Available from : `http://www.lri.fr/~magniez/PAPIERS/mag-tocs04.pdf`.

[May97] D. Mayers. Unconditionally secure quantum bit commitment is impossible. *Physical Review Letters*, 78(17) :3414–3417, 1997.

[**MMMO06**] F. Magniez, D. Mayer, M. Mosca, and H. Ollivier. Self-testing of quantum circuits. In *Proceedings of 33rd International Colloquium on Automata, Languages and Programming*, volume 4051 of *Lecture Notes in Computer Science*, pages 72–83. Verlag, 2006. Available from : `http://www.lri.fr/~magniez/PAPIERS/mmmo-icalp06.pdf`.

[**MN05**] F. Magniez and A. Nayak. Quantum complexity of testing group commutativity. In *Proceedings of 32nd International Colloquium on Automata, Languages and Programming*, volume 1770 of *Lecture Notes in Computer Science*, pages 1312–1324. Verlag, 2005. Journal version in [**MN06**].

[**MN06**] F. Magniez and A. Nayak. Quantum complexity of testing group commutativity. *Algorithmica*, 2006. Journal version of [**MN05**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/mn-algorithmica06.pdf`.

[**MNRS07**] F. Magniez, A. Nayak, J. Roland, and M. Santha. Search via quantum walk. In *Proceedings of 39th ACM Symposium on Theory of Computing*, 2007. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/mnrs-stoc07.pdf`.

[Moc04] C. Mochon. Quantum weak coin-flipping with bias of 0.192. *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pages 2–11, 2004.

[Moc05] C. Mochon. Large family of quantum weak coin-flipping protocols. *Physical Review A*, 72(022341), 2005.

[**MR04**] F. Magniez and M. de Rougemont. Property testing of regular tree languages. In *Proceedings of 31st International Colloquium on Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 932–944. Verlag, 2004. Journal version in [**MR06**].

[**MR06**] F. Magniez and M. de Rougemont. Property testing of regular tree languages. *Algorithmica*, 2006. Journal version of [**MR04**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/mr-algorithmica06.pdf`.

[MRRS04] C. Moore, D. Rockmore, A. Russell, and L. Schulman. The power of basis selection in fourier sampling : hidden subgroup problems in affine groups. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete algorithms*, pages 1113–1122, 2004.

[**MSS05**] F. Magniez, M. Santha, and M. Szegedy. Quantum algorithms for the triangle problem. In *Proceedings of 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 1109–1117, 2005. Journal version in [**MSS06**].

[**MSS06**] F. Magniez, M. Santha, and M. Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 2006. Journal version of [**MSS05**]. To appear. Available from : `http://www.lri.fr/~magniez/PAPIERS/mss-sicomp06.pdf`.

[MY98] D. Mayers and A. Yao. Quantum cryptography with imperfect apparatus. In *Proceedings of 39th IEEE Symposium on Foundations of Computer Science*, pages 503–509, 1998.

[NC00] M. Nielsen and I. Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2000.

[OSSS05] R. O'Donnell, M. Saks, O. Schramm, and R. Servedio. Every decision tree has an influential variable. In *Proceedings of 46th IEEE Symposium on Foundations of Computer Science*, pages 31–39, 2005.

[Pak00] I. Pak. Testing commutativity of a group and the power of randomization. Electronic version at `http ://www-math.mit.edu/∼pak/research.html`, 2000.

[Par66] R. Parikh. On context-free languages. *Journal of the ACM*, 13(4) :570–581, 1966.

[Pre] J. Preskill. Lecture notes for quantum computation [online]. Available from : `http://www.theory.caltech.edu/~preskill/ph229/`.

[PRR07] M. Parnas, D. Ron, and R. Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 2007. To appear. Also in TR04-010, ECCC, 2004.

[PVY02] D. Peled, M. Vardi, and M. Yannakakis. Black box checking. *Journal of Automata, Languages and Combinatorics*, 7(2) :225–246, 2002.

[RG02] T. Rudolph and L. Grover. A 2–rebit gate universal for quantum computing. Technical Report quant-ph/0210187, ArXiv, 2002.

[Ric07] P. Richter. Almost uniform sampling via quantum walks. *New Journal of Physics*, 2007. To appear.

[RS96] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing applications to program testing. *SIAM Journal on Computing*, 25(2) :252–271, 1996.

[RV76] R. Rivest and J. Vuillemin. On recognizing graph properties from adjacency matrices. *Theoretical Computer Science*, 3 :371–384, 1976.

[Sch02] U. Schöning. A probabilistic algorithm for k-SAT based on limited local search and restart. *Algorithmica*, 32(4) :615–623, 2002.

[Shi02] Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pages 513–519, 2002.

[Shi03] Y. Shi. Both Toffoli and Controlled-NOT need little help to do universal quantum computation. *Quantum Information and Computation*, 3(1) :84–92, 2003.

[Sho97] P. Shor. Algorithms for quantum computation : Discrete logarithm and factoring. *SIAM Journal on Computing*, 26(5) :1484–1509, 1997.

[SP00] P. Shor and J. Preskill. Simple proof of security of the BB84 quantum key distribution protocol. *Physical Review Letters*, 85(2) :441–444, 2000.

[SR02] R. Spekkens and T. Rudolph. A quantum protocol for cheat-sensitive weak coin flipping. *Physical Review Letters*, 89 :227901, 2002.

[SS04] M. Santha and M. Szegedy. Quantum and classical query complexities of local search are polynomially related. In *Proceedings of 36th ACM Symposium on Theory of Computing*, pages 494–501, 2004.

[ŠS06] R. Špalek and M. Szegedy. All quantum adversary methods are equivalent. *Theory of Computing*, 2(1) :1–18, 2006.

[Ste98] J. Stern. *La science du secret*. Odile Jacob, 1998.

[SV02] L. Segoufin and V. Vianu. Validating streaming xml documents. In *Proceedings of 21st ACM Symposium on Principles of Database Systems*, pages 53–64, 2002.

[Sze03] M. Szegedy. On the quantum query complexity of detecting triangles in graphs. Technical Report quant-ph/0310107, arXiv archive, 2003.

[Sze04] M. Szegedy. Quantum speed-up of markov chain based algorithms. In *Proceedings of 45th Symposium on Foundations of Computer Science*, pages 32–41, 2004.

[Tai79] K. Tai. The tree-to-tree correction problem. *Journal of the ACM*, 26 :422–433, 1979.

[Wat01] J. Watrous. Quantum algorithms for solvable groups. In *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pages 60–67, 2001.

[Yao87] A. Yao. Lower bounds to randomized algorithms for graph properties. In *Proceedings of 28th IEEE Symposium on Foundations of Computer Science*, pages 393–400, 1987.

[Yao93] A. Yao. Quantum circuit complexity. In *Proc. 34th IEEE FOCS*, pages 352–361, 1993.

[Yao94] A. Yao. Near-optimal time-space tradeoff for element distinctness. *SIAM Journal on Computing*, 23(5) :966–975, 1994.

[Yao03] A. Yao. *Personal communication*, 2003.

[YL95] Mihalis Yannakakis and David Lee. Testing finite state machines : Fault detection. *Journal of Computer and System Sciences*, 50(2) :209–227, 1995.

[Zha05] S. Zhang. On the power of Ambainis lower bounds. *Theoretical Computer Science*, 339(2-3) :241–256, 2005.

Note : Les références imprimées en caractères gras sont celles où j'apparais comme auteur.