Université de Montréal

Preuves interactives quantiques

par
Hugue Blier

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des arts et des sciences
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en informatique

Juillet, 2009

Université de Montréal
Faculté des arts et des sciences

Cette thèse intitulée:

Preuves interactives quantiques

présentée par:

Hugue Blier

a été évaluée par un jury composé des personnes suivantes:

Pierre McKenzie
président-rapporteur

Alain Tapp
directeur de recherche

Louis Salvail
membre du jury

Patrick Hayden
examinateur externe

Pierre Duchesne
représentant du doyen de la FES

Thèse acceptée le

RÉSUMÉ

Cette thèse est consacrée à la complexité basée sur le paradigme des preuves interactives. Les classes ainsi définies ont toutes en commun qu'un ou plusieurs prouveurs, infiniment puissants, tentent de convaincre un vérificateur, de puissance bornée, de l'appartenance d'un mot à un langage. Nous abordons ici le modèle classique, où les participants sont des machines de Turing, et le modèle quantique, où ceux-ci sont des circuits quantiques. La revue de littérature que comprend cette thèse s'adresse à un lecteur déjà familier avec la complexité et l'informatique quantique.

Cette thèse présente comme résultat la caractérisation de la classe NP par une classe de preuves interactives quantiques de taille logarithmique.

Les différentes classes sont présentées dans un ordre permettant d'aborder aussi facilement que possible les classes interactives. Le premier chapitre est consacré aux classes de base de la complexité ; celles-ci seront utiles pour situer les classes subséquemment présentées. Les chapitres deux et trois présentent respectivement les classes à un et à plusieurs prouveurs. La présentation du résultat ci-haut mentionné est l'objet du chapitre quatre.

Mots clés : preuves interactives, Arthur-Merlin, complexité, quantique, caractérisation.

ABSTRACT

This thesis is devoted to complexity theory based on the interactive proof paradigm. All classes defined in this way involve one or many infinitely powerful provers attempting to convince a verifier of limited power that a string belongs to a certain language. We will consider the classical model, in which the various participants are Turing machines, as well as the quantum model, in which they are quantum circuits. The literature review included in this thesis assume that the reader is familiar with the basics of complexity theory and quantum computing.

This thesis presents the original result that the class NP can be characterized by a class of quantum interactive proofs of logarithmic size.

The various classes are presented in an order that facilitates the treatment of interactive classes. The first chapter is devoted to the basic complexity classes; these will be useful points of comparison for classes presented subsequently. Chapters two and three respectively present classes with one and many provers. The presentation of the result mentioned above is the object of chapter four.

Keywords: interactive proofs, Arthur-Merlin, complexity, quantum, characterization

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES APPENDICES

LISTE DES SIGLES

$\mathsf{AM[poly]}$	*Arthur-Merlin Polynomial number of messages*
BPP	*Bounded Probabilistic Polynomial-Time*
BQP	*Bounded-Error Quantum Polynomial-Time*
EXP	*Exponential-Time*
IP	*Interactive Polynomial-Time*
MIP	*Multi-Prover Interactive Polynomial-Time*
MIP^*	*Multi-Prover Interactive Polynomial-Time with Quantum Provers*
NEXP	*Non-Deterministic Exponential-Time*
NP	*Non-Deterministic Polynomial-Time*
P	*Polynomial-Time*
PP	*Probabilistic Polynomial-Time*
$\mathsf{PQMA}_{\log}(2)$	*Polynomial-Time Quantum Arthur-Merlin two logarithmic size messages*
PSPACE	*Polynomial-space*
$\mathsf{QMA}_{\log}$	*Quantum Arthur-Merlin logarithmic size message*
$\mathsf{QMA}_{\log}(2)$	*Quantum Merlin-Arthur two logarithmic size messages*
$\mathsf{QMA}_{\log}(\sqrt{n}\mathsf{polylog(n)})$	*Quantum Merlin-Arthur* $\sqrt{n}\mathsf{polylog}(n)$ *logarithmic size messages*
$\mathsf{QAM[poly]}$	*Arthur-Merlin Polynomial number of messages*
QIP	*Quantum Interactive Polynomial-Time*
QMIP	*Quantum Multi-Prover Interactive Polynomial-Time*
$\mathsf{QMIP}_{\mathsf{le}}$	*Quantum Multi-Prover Interactive Polynomial-Time With Limited Prior Entanglement*
$\mathsf{QMIP}_{\mathsf{ne}}$	*Quantum Multi-Prover Interactive Polynomial-Time With No Prior Entanglement*

NOTATION

$\|\Psi\rangle$	Vecteur représentant l'état Ψ.
$\langle\Psi\|$	Vecteur dual de $\|\Psi\rangle$.
$\langle\Psi\|\Phi\rangle$	Produit interne de $\|\Psi\rangle$ et $\|\Phi\rangle$.

À ma mère.

REMERCIEMENTS

Une thèse est l'achèvement d'un long travail. Ce sont bien des mots qui ont dû être relus et corrigés mais aussi de longues heures de travail et de recherches, de frustrations et d'embûches, de réalisations et de satisfaction. Ce sont aussi de beaux souvenirs et de petits riens qui ont rendu le fardeau endurable. Je voudrais, pour cela, remercier ces confrères qui sont mes amis Adrien, Frédéric, Michaël et Olivier.

C'est aussi la fin d'une expérience pour laquelle je considère avoir été privilégié. On ne reste pas cinq ans sous la gouverne de son directeur sans heurts ni frictions. Mais, somme toute, j'ai adoré. Contrairement à biens d'autres, les études graduées étaient pour moi une fin en soi plutôt qu'un passage obligé. Merci Alain de m'avoir donné cette chance.

Tous ceux que j'ai remerciés sont des gens qui pourraient lire ma thèse ou mes articles et, surtout, qui ont fait les mêmes choix que moi. Mais le soutien inconditionnel de quelqu'un pour qui tout ceci n'a rien d'intéressant reste incommensurable. Et pour ceci, je peux remercier ma mère.

L'argent reste le nerf de la guerre et je voudrais mentionner la contribution de QuantumWorks (CRSNG) pour mes voyages au front, la tente sur ma tranchée et la soupe dans mon écuelle.

INTRODUCTION

Le concept de preuve est familier à quiconque aborde les sciences pures. À ce vocable est associée la certitude et à cette certitude, toute la rigidité de la science. Pourtant, en pratique, ce concept est souvent inutilisable. On veut manger santé avant de pouvoir s'expliquer en détail la chimie du corps humain. On veut que la bonne réparation soit faite sur notre voiture sans devenir mécanicien. Et l'on vote pour ces gens qui vont prendre diverses décisions dans des domaines qui nous sont étrangers. Personne n'a la capacité de tout comprendre et il faut pourtant vivre. Par comprendre, nous entendons concevoir une juste idée ; de la même manière que l'on dit apprendre. Malgré tout, les gens cherchent à vérifier.

La présente thèse n'est pas une étude anthropologique, ni épistémologique, sur la relation entre l'Homme et la certitude de la Science. Pourtant, une facette de ce besoin de vérifier une preuve mathématique peut être formalisée. Ce manque d'autonomie est le sujet de cette thèse, c'est-à-dire les preuves interactives quantiques.

Nous avons déjà mentionné la *preuve*, abordons maintenant l'*interactivité*. Une preuve, quelle qu'elle soit, est dérivée à partir d'axiomes de base. C'est plus ou moins ce que fait un chercheur. Considérons maintenant qu'un individu n'ait pas la capacité – pensons ici au temps plutôt qu'à l'intelligence, le premier pouvant pallier aux lacunes de la seconde – de construire lui-même cette preuve. Un tiers peut la lui fournir et il n'a plus qu'à la vérifier. C'est là la première étape de l'interactivité : la lecture d'une preuve dans un livre ou un article scientifique.

Mais qu'en est-il de la personne qui ne veut pas consacrer le temps nécessaire à la compréhension d'une preuve ? Prenons un exemple plus concret : un cancre et un élève studieux font équipe pour un devoir de mathématique. Le cancre n'a bien sûr pas travaillé alors que c'est l'élève studieux qui a rédigé le devoir. Le cancre veut tout de même s'assurer qu'il aura une bonne note. Il parcourt donc la preuve fournie par l'élève studieux en lui posant des questions de-ci de-là. S'il obtient des réponses

convaincantes à chacune des questions, en posant suffisamment de questions, il sera rassuré.

La preuve n'est plus alors une suite d'axiomes, mais seulement une argumentation qui ne convainc que celui qui pose les questions. Il s'agit là d'une preuve interactive.

La preuve ici décrite était pure information, de l'information que s'échangent deux participants. Mais il est aussi possible de considérer un objet comme une preuve. Les participants peuvent alors s'échanger de la matière, avec toutes les implications que la théorie permet. On peut alors l'aborder au niveau *quantique*. De plus, l'aléa est fondamental dans une preuve interactive, il l'est aussi dans la mécanique quantique. Les deux semblent donc se marier particulièrement bien et le résultat en est les preuves interactives quantiques.

Structure du document

Le sujet des preuves interactives est ce sujet très important qui est bien souvent le dernier traité dans un cours d'introduction à l'informatique théorique. Et malheureusement, le temps de définir les différents modèles de machines de Turing, les circuits, les autres modèles de calculs, les langages, les différents types de réduction... la session risque d'être terminée. S'il reste une heure ou deux, on peut alors présenter la preuve de Shamir que $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92] et laisser les étudiants ébaubis.

Nous supposons donc que le lecteur est déjà familier avec les concepts de base de l'informatique théorique et quantique. Ceci correspond au corpus du cours d'introduction à l'informatique théorique et à celui du cours d'informatique quantique de l'Université de Montréal. Ceci nous permet d'aborder aussi directement que possible les preuves interactives quantiques. Le lecteur est invité à consulter les livres suivants pour toute notion ici utilisée qui lui serait étrangère.

1. *Introduction to Automata Theory, Languages, and Computation*, J. Hopcroft et J. Ullman, Addison-Wesley, 1979

2. *Quantum Computation and Quantum Information* M. Nielsen et I. Chuang, Cambridge University Press, 2002

Les trois premiers chapitres sont consacrés à la revue de littérature. Afin qu'elle soit cohérente, des notions de bases ainsi que des résultats de complexité classique sont présentés. Le premier chapitre introduit les classes de complexité qui ne sont pas interactives et auxquelles seront comparées les classes subséquemment définies. Les chapitres deux et trois présentent respectivement les classes interactives définies à l'aide d'un et de plusieurs Prouveur. Finalement, le chapitre quatre présente l'un des résultats de recherche du candidat.

Cette présentation ne respecte pas l'ordre traditionnel qui est de d'abord présenter les résultats classiques et, ensuite, les résultats quantiques. Les preuves interactives se distinguant naturellement par le nombre de Prouveur, il nous a semblé plus propice d'utiliser cette séparation. Cependant, chaque chapitre est séparé afin de présenter d'abord les résultats classiques suivis des résultats quantiques.

Contributions du candidat

Le candidat a, au cours de ses études doctorales, publié trois articles ; ceux-ci sont brièvement décrits dans la présente section. Sa thèse est aussi à considérer. Elle comporte tout d'abord une revue de littérature approfondie présentant de manière originale l'ensemble du domaine. Ensuite, elle contient la présentation détaillée du dernier article précédemment mentionné.

Multiparti

Cet article [BT08] est étranger au sujet de cette thèse. Il propose un modèle de calcul multiparti dans lequel un serveur peut fournir des ressources aux acteurs d'un calcul multiparti. Le calcul est alors sécuritaire au sens de la théorie de l'information. Il a été montré qu'il est alors possible de faire une mise en gage, de faire des évaluations sur des mises en gage ainsi que de faire du calcul multiparti. La seule condition de sécurité est qu'il n'y ait pas de collusion avec le serveur. Le nombre de participants malhonnêtes peut être arbitraire et même le serveur peut être mal intentionné. Ce travail n'utilise pas la mécanique quantique.

L'idée originale de calcul à deux participants vient de son directeur. La contribution du candidat à cet article a été la généralisation à plusieurs participants, la formalisation des protocoles, ainsi que la définition de sécurité. Celle-ci est originale en ceci qu'elle ne se base pas sur un ratio de participants malhonnêtes mais sur l'interaction d'un seul des participants. La sécurité étant ainsi définie, ces résultats s'avèrent avoir des applications pratiques, en particulier dans tout ce qui a trait aux services de partage d'information sur internet.

Preuves courtes

Chris Marriott et John Watrous ont montré que $\mathsf{QMA}_{\log} = \mathsf{BQP}$ [MW05]. En d'autres termes, BQP caractérise la classe analogue à QMA où le message est de taille logarithmique. Nous avons montré dans cet article [BT09a] qu'étant donnée une promesse supplémentaire sur l'intrication à l'intérieur du message, nous pouvions définir une classe semblable à $\mathsf{QMA}_{\log}$ incluant NP. Ceci met en lumière la différence entre l'information quantique et l'information classique ainsi que le rôle de l'intrication à l'intérieur de preuves interactives. Cette classe, $\mathsf{QMA}_{\log}(2)$, a, par contre, une résilience polynomialement proche de 1.

Ce résultat a d'intéressant qu'il présente un rapprochement entre BPP et NP.

En effet, ces deux classes n'ont pas de relations connues et il est intéressant de tenter de définir une classe – en cherchant à ce qu'elle soit aussi *petite* que possible – qui puisse les inclure toutes deux.

La contribution du candidat à cet article a été la formalisation du protocole, la preuve du résultat ainsi que la majeure partie de la rédaction de l'article.

Caractérisation quantique de NP

Dans [BT09b], nous avons proposé une classe dans le paradigme des preuves interactives quantiques de taille logarithmique qui caractérise complètement la classe NP. Contrairement au travail effectué dans [BT09a] – donnant un rapprochement entre BQP et NP –, nous nous sommes ici consacrés à la classe NP. C'est un des plus grands défis de la complexité que de rapprocher autant que possible le paradigme de l'informatique classique de celui de l'informatique quantique. Il est donc d'un grand intérêt de caractériser complètement une classe aussi fondamentale que NP. Ceci met en relief les propriétés de la mécanique quantique dissociant les preuves interactives quantiques de celles classiques, en particulier le rôle de l'intrication à l'intérieur du message. Ce résultat met donc en lumière ce qui pourrait être à la base de la distinction entre QMA et NP.

L'idée originale de la définition de la classe ainsi que la preuve d'égalité sont l'œuvre du candidat.

Le quatrième chapitre est entièrement consacré à la présentation de ce résultat.

CHAPITRE 1

EN SOLITAIRE

Converser de preuves interactives donne rapidement naissance à des personnages, des personnalités et des rivalités. Mais derrière cette mythologie se cache tout de même un rigoureux formalisme. Même s'il sera bientôt question d'Arthur qui communique avec Merlin, nous définirons d'abord formellement le contexte étudié.

Il sera question dans le présent chapitre de classes de complexité définies à partir de machines de Turing classiques et de circuits quantiques. Même si la plupart des modèles de machines sont équivalents du présent point de vue, le lecteur est invité à ne considérer que le modèle le plus simple : celui ayant un seul ruban infini à droite.

Nous débutons par la présentation des classes qui ne sont pas interactives. Le présent chapitre y est entièrement consacré. Il est certain que les classes définies à l'aide d'interaction ne vivent pas dans un monde séparé du reste de la complexité du calcul ; chaque classe interactive doit être le plus possible considérée par rapport aux autres classes. Nous définissons ici les classes les plus courantes, celles qui nous seront utiles afin de situer celles interactives.

1.1 Classes déterministes

Les principales mesures de complexité ne changent pas lorsque l'on considère les preuves interactives, il s'agit toujours du temps et de l'espace. Nous définissons donc d'abord, en toute généralité, ce qu'est une classe déterministe dont la mesure de complexité est le temps.

Définition 1 ($\mathsf{TIME}(\mathsf{t}(\mathsf{n}))$)**.** *Un langage L est dans la classe* $\mathsf{TIME}(\mathsf{t}(\mathsf{n}))$ *s'il existe une machine de Turing déterministe décidant le langage L en $O(t(n))$ transitions où n est la taille du mot considéré.*

Cette classe générique nous permettra d'abord de définir ce qui sera notre classe de base. Du point de vue qui nous intéresse, cette classe sera considérée comme la classe des affirmations que l'on peut prouver de façon *efficace*. Il est en effet courant, même si quelque peu arbitraire, de considérer comme efficace un algorithme fonctionnant en temps polynomial par rapport à la taille de l'entrée.

Définition 2 (P)**.** *La classe* Polynomial-Time *est*

$$\mathsf{P} = \bigcup_{c \in \mathbb{N}} \mathsf{TIME}(\mathsf{n}^{\mathsf{c}}).$$

De la même façon, nous pouvons définir n'importe quelle classe à l'aide d'une famille de fonctions. En particulier, nous serons intéressé par l'ensemble des fonctions exponentielles.

Définition 3 (EXP)**.** *La classe* Exponential-Time *est*

$$\mathsf{EXP} = \bigcup_{c \in \mathbb{N}} \mathsf{TIME}(2^{\mathsf{n}^{\mathsf{c}}}).$$

Cas particulièrement rare en complexité, nous savons démontrer que les deux classes précédentes sont distinctes. En effet, il est trivial de voir que $\mathsf{P} \subseteq \mathsf{EXP}$. Par contre, il est à première vue possible que les deux classes soient en fait égales ! Il est intuitivement clair qu'une machine de Turing pouvant fonctionner en temps exponentiel est plus *puissante* qu'une machine fonctionnant en temps polynomial. Pourtant, on peut rarement arriver à confirmer notre intuition. C'est ici rendu possible grâce au théorème suivant :

Théorème 1 (Hiérarchie temporelle)**.** *Pour toute fonction constructible en temps $t(n)$, il existe un langage L décidable en temps $O(t(n))$ mais non en temps $o\left(\frac{t(n)}{\log t(n)}\right)$.*

Corollaire 1. $\mathsf{P} \subsetneq \mathsf{EXP}$.

Tout comme nous avons défini la classe $\mathsf{TIME}(\mathsf{t}(\mathsf{n}))$, nous pouvons définir la classe $\mathsf{SPACE}(\mathsf{s}(\mathsf{n}))$.

Définition 4 ($\mathsf{SPACE}(\mathsf{s}(\mathsf{n}))$)**.** *Un langage L est dans la classe $\mathsf{SPACE}(\mathsf{s}(\mathsf{n}))$ s'il existe une machine de Turing déterministe pouvant décider le langage L en utilisant $O(s(n))$ cellules sur son ruban, où n est la taille du mot considéré.*

Et de façon similaire à P, nous pouvons définir l'ensemble des langages décidables en *espace* polynomial.

Définition 5 (PSPACE)**.** *La classe* Polynomial-space *est*

$$\mathsf{PSPACE} = \bigcup_{c \in \mathbb{N}} \mathsf{SPACE}(\mathsf{n}^{\mathsf{c}}).$$

Nous pouvons situer cette classe par rapport aux deux premières classes définies. Bien que les inclusions soient triviales, nous ne savons pas distinguer PSPACE de P ou EXP. Il y a pourtant forcément une des deux inclusions qui est stricte ; ceci dû au théorème 1. De façon générale, comparer le temps et l'espace comme mesure de complexité est une des plus grandes difficultés de la discipline.

Théorème 2. $\mathsf{P} \subseteq \mathsf{PSPACE} \subseteq \mathsf{EXP}$.

1.2 Classes non-déterministes

C'est ici que l'on peut commencer à parler réellement de preuves interactives. Le non-déterminisme a quelque chose de *magique*, caractéristique que l'on attribuera à certains participants du prochain chapitre. La naissance même des preuves interactives vient d'une généralisation du non-déterminisme [GMR85].

Définition 6 ($\mathsf{NTIME}(\mathsf{t}(\mathsf{n}))$)**.** *Un langage L est dans la classe* $\mathsf{NTIME}(\mathsf{t}(\mathsf{n}))$ *s'il existe une machine de Turing non-déterministe décidant le langage L en $O(t(n))$ transitions, où n est la taille du mot considéré.*

Nous pouvons alors définir l'ensemble des langages pouvant être décidés efficacement de façon non-déterministe.

Définition 7 (NP)**.** *La classe* Non-Deterministic Polynomial-Time *est*

$$\mathsf{NP} = \bigcup_{c \in \mathbb{N}} \mathsf{NTIME}(\mathsf{n}^{\mathsf{c}}).$$

Mais décider d'un langage de façon non-déterministe est si peu réaliste que le concept d'*efficacité* perd tout son sens. C'est là qu'intervient l'interprétation la plus répandue de la classe NP et notre introduction aux preuves interactives. Nous dirons que NP est l'ensemble des affirmations pouvant être *vérifiées* efficacement. Ceci permet de la comparer à P.

Souvenons-nous de ce qui distingue une machine non-déterministe d'une machine déterministe : dans une certaine configuration, une machine non-déterministe peut faire face à plus d'une transition possible. Considérons maintenant que sur un ruban distinct, en lecture seule, soit fournie à une machine de Turing la suite des transitions menant à l'état acceptant. C'est ce que l'on nomme parfois le *certificat*. Une telle machine déterministe n'aurait alors qu'à vérifier que chacune des transitions est valide et à accepter le mot si et seulement si un tel certificat existe. À un niveau d'abstraction plus élevé, on peut considérer ce certificat comme une preuve de l'affirmation, preuve qui nous serait fournie par un tiers.

Avec cette formalisation, les affirmations que l'on peut efficacement vérifier se placent tout naturellement entre celles que l'on peut prouver efficacement en temps et celles que l'on peut prouver efficacement en espace – bien que la notion d'efficacité en espace soit ici moins naturelle.

Théorème 3. $\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE}$.

Et comme nous avons défini EXP similairement à P, nous définissons ici NEXP similairement à NP.

Définition 8 (NEXP)**.** *La classe* Non-Deterministic Exponential-Time *est*

$$\mathsf{NEXP} = \bigcup_{c \in \mathbb{N}} \mathsf{NTIME}(2^{n^c}).$$

Il nous serait possible de sans cesse définir des classes de plus en plus grandes. Mais pour l'analyse des preuves interactives, NEXP règnera au sommet. Nous ne donnons donc dans le théorème suivant qu'une borne inférieure à la classe.

Théorème 4. $\mathsf{EXP} \subseteq \mathsf{NEXP}$.

En parallèle aux classes non-déterministes définies par le temps (i.e. le nombre de transitions), il serait naturel de mentionner celles définies par l'espace.

Définition 9 (NSPACE(s(n)))**.** *Un langage L est dans la classe* NSPACE(s(n)) *s'il existe une machine de Turing non-déterministe pouvant décider le langage L en utilisant $O(s(n))$ cellules sur son ruban, où n est la taille du mot considéré.*

Or le théorème suivant permet de situer directement toutes les classes ainsi définies, du moins dans la mesure de nos besoins.

Théorème 5 (Théorème de Savitch)**.** *Pour toute fonction $s(n) \geq \log(n)$, où n est la taille de l'entrée,* $\mathsf{NSPACE}(s(n)) \subseteq \mathsf{SPACE}(s(n)^2)$.

1.3 Classes probabilistes

Si le non-déterminisme est inutilisable en pratique, il existe tout de même une ressource fort utile qui, elle, est utilisée à profusion. Elle l'est avec raison puisqu'à ce jour (nous formaliserons cette affirmation plus loin) elle permet, en pratique, une plus grande efficacité pour certains problèmes ; cette efficacité se fait par contre au prix de devoir accepter une certaine marge d'erreur. Cette ressource est l'aléa. En effet, il est commun d'entendre parler d'une heuristique qui fonctionne avec une *bonne probabilité*. Malgré tout, il est envisageable que le gain ne soit pas plus grand que polynomial. Mais ceci n'a pas été démontré.

De façon générale, on peut penser à un algorithme probabiliste comme à une machine de Turing probabiliste. Rappelons qu'une machine probabiliste est une machine déterministe utilisant un ruban spécial de bits aléatoires (en lecture seule) ; ces bits étant considérés dans la fonction de transition. Cet algorithme sera donc juste, ou pas, selon cette liste de bits. Ceci peut bien sûr être formalisé, il s'agit de l'opérateur BP.

Définition 10 (Opérateur BP)**.** *L'opérateur* BP *(Bounded Probabilistic) est un opérateur unaire s'appliquant à une classe de complexité. Soit une classe de complexité C. Un langage L est tel que $L \in \mathsf{BP} \cdot C$ s'il existe une machine de Turing déterministe décidant d'un langage $L' \in C$ et un polynome p tels que*

- $\forall x \in L$, *au moins* $\frac{2}{3}$ *des chaînes* y *de taille* $p(|x|)$ *sont telles que* $\langle x, y\rangle \in L'$.
- $\forall x \notin L$, *au plus* $\frac{1}{3}$ *des chaînes* y *de taille* $p(|x|)$ *sont telles que* $\langle x, y\rangle \in L'$.

Ceci correspond à l'intuition : en ajoutant une composante aléatoire (la chaîne y) il est possible de décider si un mot (x) est dans le langage (L) avec une bonne probabilité ($\frac{2}{3}$). L'exemple le plus connu est la classe BPP. Étant donné que nous avons défini l'opérateur BP et la classe P, il nous suffirait pour définir BPP de dire

que $\mathsf{BPP} = \mathsf{BP} \cdot \mathsf{P}$. Mais, pour des raisons didactiques, nous définissons BPP de façon équivalente avec une machine de Turing non-déterministe.

Définition 11 (BPP)**.** *Un langage L est dans la classe BPP (Bounded Probabilistic Polynomial-Time) s'il existe une machine de Turing non-déterministe et un polynôme t tels que :*

- *Si $x \in L$ alors au moins $\frac{2}{3}$ des chemins d'exécution mènent à l'état acceptant en temps $O(t(|x|))$.*
- *Si $x \notin L$ alors au plus $\frac{1}{3}$ des chemins d'exécution mènent à l'état acceptant en temps $O(t(|x|))$.*

Le lecteur aura remarqué que nous n'avons pas défini cette classe comme nous avons défini les précédentes. Nous aurions, en effet, pu définir une classe correspondant à un temps précis sous le paradigme probabiliste borné puis définir BPP comme l'union de ces classes. Nous avons plutôt choisi de spécifier directement l'existence d'un polynôme dont nous considérons l'ordre. Ces deux façons de faire sont évidemment équivalentes ; la dernière étant plus concise et plus courante. De plus, contrairement à ce que nous avons fait précédemment, ces classes préalablement définies se seraient avérées inutiles.

Théorème 6. $\mathsf{P} \subseteq \mathsf{BPP} \subseteq \mathsf{PSPACE}$.

La classe BPP a cette particularité que la probabilité d'erreur peut être réduite arbitrairement simplement en répétant l'exécution d'une machine de Turing probabiliste un nombre polynomial de fois et en considérant la majorité des réponses comme réponse finale. Ceci est possible du fait qu'il y ait une différence constante entre la probabilité d'accepter le mot dépendamment qu'il soit effectivement dans le langage ou non. C'est ce que signifie *bounded*. Mais il est aussi possible de définir l'ensemble des langages où une telle distinction n'est pas nécessaire.

Définition 12 (PP)**.** *Un langage L est dans la classe* PP *(Probabilistic Polynomial-Time) s'il existe une machine de Turing non-déterministe et un polynôme t tels que :*

- *Si $x \in L$ alors au moins la moitié ($\frac{1}{2}$) de ces chemins d'exécution mènent à l'état acceptant en temps $O(t(|x|))$.*
- *Si $x \notin L$ alors strictement moins de la moitié ($\frac{1}{2}$) de ces chemins d'exécution mènent à l'état acceptant en temps $O(t(|x|))$.*

Naturellement, tout langage dans BPP est par définition dans PP et pour décider d'un langage dans PP il suffit de faire le compte des chemins acceptant en temps $O(t(|x|))$, ce qui se fait en espace polynomial. Nous pouvons ainsi situer cette classe.

Théorème 7. $\mathsf{BPP} \subseteq \mathsf{PP} \subseteq \mathsf{PSPACE}$.

Définition 13 ($\mathsf{BP} \cdot \mathsf{PP}$)**.** *La classe* $\mathsf{BP} \cdot \mathsf{PP}$ *correspond à l'opérateur* BP *appliqué à la classe* PP*.*

La définition précédente est plutôt prosaïque. Mais détailler formellement l'effet de l'opérateur ne sert à rien. Intuitivement cela signifie que, pour les langages contenus dans $\mathsf{BP} \cdot \mathsf{PP}$, au moins deux tiers des chaînes aléatoires sont telles qu'au moins la moitié des chemins d'exécution seront justes.

Il est clair que $\mathsf{PP} \subseteq \mathsf{BP} \cdot \mathsf{PP}$. Et encore une fois, il est possible de faire le compte des chemins d'exécution afin de décider d'un langage dans $\mathsf{BP} \cdot \mathsf{PP}$. Et donc :

Théorème 8. $\mathsf{PP} \subseteq \mathsf{BP} \cdot \mathsf{PP} \subseteq \mathsf{PSPACE}$.

1.4 Classes quantiques

Étant donnés les résultats qui seront présentés au chapitre 4, il est naturel de s'intéresser particulièrement aux classes quantiques. Un néophyte remarquera que la très grande majorité des classes quantiques correspond à la *traduction* quantique de classes classiques. Ceci vient d'une motivation légitime : comparer la puissance de l'ordinateur quantique à celle de l'ordinateur classique. Et malheureusement, force est de constater que les résultats sont plutôt modestes. Ceci était prévisible tant nous avons peine à séparer les classes classiques entre elles.

Nous avons utilisé le modèle des machines de Turing afin de définir les classes précédentes. De manière tout à fait équivalente, nous aurions pu considérer une machine de Turing quantique [Deu85] afin de définir les classes quantiques. Mais ce modèle est à ce jour peu utilisé et on lui préfère le modèle des familles uniformes de circuits quantiques . Nous considérerons ici un ensemble de portes universel tel que, Hadamard (H), $\frac{\pi}{8}$ (T) et CNOT. Le lecteur pourra considérer un ensemble de portes universel équivalent s'il le souhaite.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{pmatrix}, \quad CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Comme nous ne considérons ici que les problèmes de décision, nous supposerons, sans perte de généralité, que le circuit prend une entrée classique et a une sortie classique. C'est à dire que le circuit C sur entrée x (que nous notons $C(x)$) débute avec un encodage dans un registre quantique de l'entrée. Puis, à la toute fin du circuit, une mesure dans la base de calcul est faite sur le premier qubit de sortie. Le résultat étant $y \in \{0, 1\}$, nous noterons ceci $C(x) = y$ et considérons 1 comme l'acceptation du mot x et 0 comme son rejet.

Tout comme l'on considère que les problèmes résolubles par un ordinateur classique sont les problèmes pouvant être résolus en temps polynomial par rapport à la taille du problème, on considérera qu'est à la portée de l'ordinateur quantique ce qui peut être fait en temps polynomial par un ordinateur quantique. Dans le cas classique, deux classes correspondent à cette considération : P et BPP. La mécanique quantique étant probabiliste en son essence, on considère seulement la classe quantique correspondant à BPP.

Définition 14 (BQP)**.** *Un langage L est dans la classe* BQP *(Bounded-Error Quantum Polynomial-Time) s'il existe un polynôme t et une famille uniforme de circuits quantiques $\mathcal{C} = \{C_n\}_{n>0}$ de taille $O(t(n))$ tels que pour tout mot x de taille n :*

- *Si $x \in L$ alors $C_n(x) = 1$ avec probabilité au moins $\frac{2}{3}$.*
- *Si $x \notin L$ alors $C_n(x) = 1$ avec probabilité au plus $\frac{1}{3}$.*

Bien sûr, puisque BQP n'est que la version quantique de BPP, il est direct de réaliser que la première classe inclut la seconde. Pour ce qui est de la seconde inclusion du théorème suivant, nous invitons le lecteur à lire une preuve simplifiée dans [MW05]. Il s'agit en fait d'une preuve que QMA $\subseteq$ PP. Nous définirons dans le prochain chapitre la classe QMA ; qu'il suffise pour l'instant de dire que QMA $\subseteq$ PP, d'où le résultat qui nous intéresse.

Théorème 9. BPP $\subseteq$ BQP $\subseteq$ PP*.*

Tout comme nous avons fait l'analyse des classes classiques basées sur temps, il nous reste à aborder les classes quantiques basées sur l'espace. Ici, l'espace sera mesuré en nombre de qubits nécessaires. Mais la plus grande différence vient de la nature de la mécanique quantique elle-même : l'incertitude. Alors que l'espace classique peut être déterministe, la contrepartie quantique se doit d'être considérée comme probabiliste. Nous définissons donc QSPACE(s(n)) avec la plus faible restriction possible (tout en restant maniable) quant à ces probabilités.

Définition 15 (QSPACE(s(n))). *Un langage L est dans la classe* QSPACE(s(n)) *(Quantum Space) s'il existe une fonction constructible en espace s et et une famille uniforme de circuits quantiques $\mathcal{C} = \{C_n\}_{n>0}$ de largeur $O(s(n))$ tels que pour tout mot x de taille n :*

- *Si $x \in L$ alors $C(x) = 1$ avec probabilité au moins $\frac{1}{2} + \varepsilon$.*
- *Si $x \notin L$ alors $C(x) = 1$ avec probabilité au plus $\frac{1}{2}$.*

Comment alors se comparent SPACE(s(n)) et QSPACE(s(n)) ? John Watrous a répondu à cette question de manière très élégante :

Théorème 10. *[Wat03] Soit $s(n) \in \Omega(\log(n))$, une fonction constructible en espace, alors* $\mathsf{QSPACE}(\mathsf{s}(\mathsf{n})) \subseteq \mathsf{SPACE}(\mathsf{s}^2(\mathsf{n}))$

Si nous définissons QPSPACE de manière analogue à PSPACE nous obtenons le lemme suivant :

Définition 16 (QPSPACE). *La classe* Quantum Polynomial Space *est*

$$\mathsf{QPSPACE} = \bigcup_{c \in \mathbb{N}} \mathsf{QSPACE}(\mathsf{n}^c).$$

Lemme 1. $\mathsf{QPSPACE} = \mathsf{PSPACE}$.

1.5 Questions

Tel qu'annoncé dans l'introduction, nous mettons ici en relief l'étendue de notre ignorance. En effet, à la lecture de ce chapitre, le lecteur aura plus de questions que de réponses et nous voulons le réconforter un peu. Beaucoup d'inclusions sont connues parmi les classes de complexité, mais ce sont les égalités et les inégalités qui sont les plus difficiles à atteindre. Nous présentons ici brièvement les différentes grandes questions concernant les classes jusqu'ici présentées et les croyances qui y sont rattachées.

La place d'honneur revient sans conteste à cette question :

Question 1. P *vs* NP.

C'est la question qui promet richesse et gloire à celui qui y répondra. Un peu paradoxalement, nombre de gens ont prétendu, ou prétendent encore, avoir répondu à la question, parfois en affirmant que $\mathsf{P} = \mathsf{NP}$ ou alors que $\mathsf{P} \neq \mathsf{NP}$. Après avoir trouvé bien des erreurs dans les premières preuves présentées, les chercheurs ne se bâdrent même plus à lire la plupart des propositions. Il faudra donc, à celui qui aura la réponse, beaucoup de pouvoir de persuasion pour être écouté. La croyance répandue est que $\mathsf{P} \neq \mathsf{NP}$. Et une preuve de ce fait aura beaucoup plus de chances d'être considérée.

Mais tout n'est pas perdu pour celui qui aura prouvé que $\mathsf{P} = \mathsf{NP}$. En particulier, si sa preuve s'avère constructive, il pourra aboutir à un algorithme en temps polynomial pour un problème pratique tel que la factorisation. Si l'algorithme est efficace (i.e. le degré et les constantes du polynôme sont petits) pouvoir factoriser de grands nombres, et de ce fait briser toute la sécurité de RSA, est un moyen très persuasif d'être écouté. Briser RSA – ce problème n'est pas NP-ardu – n'est pas une preuve mais peut tout de même s'avérer un peu dangereux.

Au-delà des romans d'espionnage, de la gloire et des considérations pécunières, ceci est peut-être la plus grande question sur la nature même des mathématiques. Comme nous l'avons présenté, P étant ce qui peut être prouvé efficacement et NP ce qui peut être vérifié efficacement, prouver leur relation nous donne un aperçu de la nature même d'une preuve mathématique. Il ne faut pas non plus oublier qu'une troisième réponse existe : il se pourrait qu'il soit simplement impossible de répondre à la question, et que l'on prouve cette impossibilité. Plus précisément, il se pourrait que leur relation soit indémontrable à l'intérieur de l'arithmétique de Peano.

Suit une question que nous trouvons malgré tout plus intéressante :

Question 2. P *vs* PSPACE.

Le plus probable est que $\mathsf{P} \neq \mathsf{PSPACE}$ et même la possibilité qu'il soit impossible de répondre à la question reste difficilement envisageable. Or, puisque le temps et l'espace sont les deux mesures de complexité fondamentales, il semble être d'un grand intérêt de savoir si elles sont équivalentes ou irréconciliables. Il se pourrait toutefois que cette question soit elle aussi sans réponse.

La question suivante est elle aussi d'un grand intérêt.

Question 3. NP *vs* BPP.

On peut voir ces deux classes comme des généralisations de P. Il serait alors intéressant de savoir comment se comparent ces deux paradigmes. Or, à ce jour, il n'y a que très peu de résultats ou même de prise de position sur le sujet. Nous croyons que s'il est difficile de comparer le déterminisme avec l'un ou l'autre du non-déterminisme et du probabilisme, comparer ces deux derniers est une bien plus lourde tâche. Mais ceci n'est qu'une impression.

Finalement l'une des grandes questions, particulièrement pour des centaines de physiciens qui travaillent sur la construction d'un ordinateur quantique :

Question 4. BPP *vs* BQP.

En pratique, ce qui peut être fait par un ordinateur classique est représenté par la classe BPP et ce qui pourra être fait par un ordinateur quantique, par la classe BQP. Il est donc d'un grand intérêt de savoir si l'ordinateur quantique est effectivement plus puissant que l'ordinateur classique. Ou, plus précisément, si le gain en temps est plus grand que polynomial. Cette question n'a, à ce jour, pas de réponse et la croyance est que $\mathsf{BPP} \neq \mathsf{BQP}$, mais la plupart des gens sont ici juges et parties. Même si ce ne serait pas la fin de l'informatique quantique – il resterait par exemple la distribution de clefs –, démontrer que $\mathsf{BQP} = \mathsf{BPP}$ amoindrirait substantiellement les perspectives du domaine.

CHAPITRE 2

EN COUPLE

Suite au chapitre précédent, nous sommes outillé pour aborder les preuves interactives à proprement parler. Comme dit précédemment, nous définissons les différentes classes interactives en utilisant certains *personnages* tels que le Prouveur et le Vérificateur, ce qui peut donner l'impression d'un manque de rigueur alors qu'il n'en est rien. Nous débuterons donc ce chapitre en définissant le modèle d'interaction ainsi que deux notions fondamentales.

Mais d'abord, commençons par un exemple. Au chapitre précédent, nous avons mentionné que NP peut être vu comme une classe interactive. Nous avons aussi mentionné que l'ensemble des choix non-déterministes, que nous avons nommé *certificat*, peut être considéré comme une preuve. Pour rendre tout ceci plus tangible, considérons le langage qui est l'ensemble des paires de graphes isomorphes (langage bien connu sous le nom *graph isomorphism*) et supposons que face à deux graphes, quelqu'un veuille vérifier que ceux-ci sont isomorphes. Nous nommerons cette personne le Vérificateur. Il n'y a pas à ce jour d'algorithme connu pour décider en temps polynomial si deux graphes sont isomorphes ; nous considérons donc que le Vérificateur ne peut pas par lui-même décider si cette paire fait partie du langage. Quelqu'un connaissant l'isomorphisme entre les deux graphes pourrait le prouver au Vérificateur. Il s'agit donc du Prouveur. De manière générale, nous considérons le Prouveur comme étant tout-puissant ; il est donc toujours en mesure d'user de la stratégie optimale pour convaincre le Vérificateur. Dans le cas présent, celui-ci n'a qu'à donner une description de l'isomorphisme au Vérificateur. Ce dernier vérifie que l'isomorphisme est valide (en vérifiant chaque voisin de chaque nœud) et peut donc déclarer si deux graphes sont isomorphes.

Le modèle d'interaction entre deux machines de Turing généralise l'exemple

précédent. Il y a interaction entre deux machines A et B lorsque, en plus de leurs rubans respectifs, il existe deux rubans a et b où a est en lecture seule pour A et en écriture seule pour B et b est en lecture seule pour B et en écriture seule pour A. Ainsi, l'isomorphisme de l'exemple précédent serait transmis sur le ruban en lecture seule pour le Vérificateur et un autre ruban serait inutilisé. Dorénavant, nous considérerons que les machines interagissent sans le mentionner explicitement.

Suite à l'interaction, le Vérificateur acceptera ou non le mot. Dans l'exemple précédent, si deux graphes sont isomorphes, la probabilité du Prouveur de convaincre le Vérificateur est 1 et si deux graphes ne sont pas isomorphes, cette probabilité est nulle. Or, il n'en est pas toujours ainsi. Les classes interactives sont par nature probabilistes et nous aurons besoin des deux définitions suivantes pour bien les décrire.

On voudra d'abord accepter avec une bonne probabilité les mots qui appartiennent au langage.

Définition 17 (Complétude)**.** *Soit L un langage et V le Vérificateur. On dira que la complétude de la machine de Turing probabiliste V pour le langage L est $c(|x|)$ si $\exists$ P, un Prouveur, tel que $\forall x \in L$ la probabilité qu'a la machine V d'accepter le mot x est bornée inférieurement par $c(|x|)$.*

Et tout comme il est important de ne pas rejeter un mot qui appartient au langage, il n'est pas désirable d'accepter un mot qui n'y appartient pas.

Définition 18 (Résilience)**.** *Soit L un langage et V le Vérificateur. On dira que la résilience de la machine de Turing probabiliste V pour le langage L est $r(|x|)$ si $\forall P$ et $\forall x \notin L$ la probabilité qu'a la machine V d'accepter le mot x est bornée supérieurement par $r(|x|)$.*

2.1 Classique

Les preuves interactives et la classe qui s'y rattache ont été formellement définies par Goldwasser, Micali et Rackoff [GMR85]. S'intéressant à la nature d'une preuve, ils cherchaient à quantifier l'information que devait nécessairement acquérir quelqu'un pour être convaincu d'un énoncé. Non seulement cela a-t-il donné naissance à IP mais aussi à la classe ZK (*Zero-Knowledge*) dont nous ne traiterons pas ici. Ils ont d'abord représenté la classe NP tel qu'illustré à la figure 2.1. Ici, le Prouveur est nommé A et peut écrire (write) sur le ruban d'interaction. Le Vérificateur est nommé B et ne peut que lire (read) ce ruban.

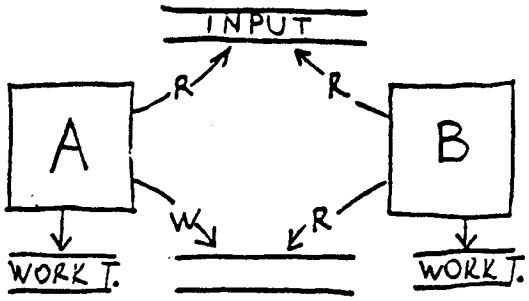

Fig. 1: The *NP* proof-system(*)

FIGURE 2.1 – Diagramme original de [GMR85]

Ils ont ensuite eu cette intuition : une preuve rendue dans un livre ou un article est beaucoup plus contraignante puisque l'on doit répondre à toutes les questions possibles à l'avance. Au contraire, lors d'un cours, les étudiants, même s'ils ne sont pas certains de tous les détails, ont la possibilité de poser des questions et, au final, d'être convaincus de la véracité d'un énoncé. Ceci a mené à la modélisation de la généralisation de NP telle qu'illustrée à la figure 2.2.

Il existe un exemple très répandu de preuves interactives : le test à l'aveugle. Si un fabricant de détergeant tente de vous convaincre que le produit de marque

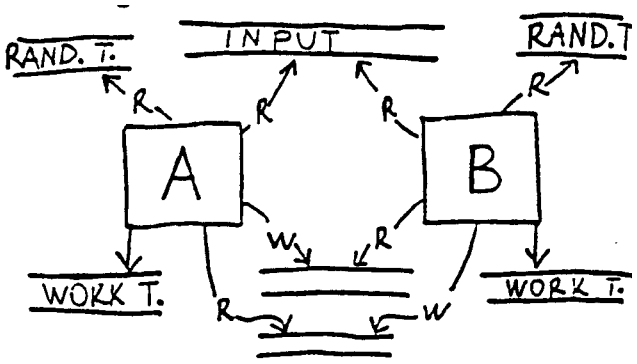

Fig. 2: an interactive pair of Turing machines

FIGURE 2.2 – Diagramme original de [GMR85]

maison n'est pas exactement le même que le sien, vous pouvez lui proposer un test. Ce test est bien sûr inutile si la différence est évidente, telle que la couleur. Vous tirez à pile ou face pour choisir l'un des deux détergents. Puis vous demandez au fabricant de dire duquel des deux il s'agit. Il est certain de réussir si effectivement ils sont différents alors que s'ils sont identiques, il n'a qu'une chance sur deux de réussir. C'est là l'esprit des preuves interactives.

Définition 19 (IP)**.** *Un langage L est dans la classe* IP *(Interactive Polynomial-Time) s'il existe un Vérificateur et un polynôme t tel que, pour un mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Nous tenons à mettre l'emphase sur le fait que IP signifie *Interactive Polynomial-Time* et non pas *Interactive Proof*. Cette erreur très répandue est une aberration tout autant que l'est d'assimiler NP à *Non-Polynomial*. Le *Polynomial-Time* réfère à la contrainte sur le Vérificateur que l'on restreint polynomialement en temps.

Les complétude et résilience semblent tout à fait arbitraires. Ne serait-il pas possible de vouloir être *plus convaincu* que $\frac{2}{3}$, si l'on peut dire ? Dans l'exemple précédent du test à l'aveugle, il est possible de répéter le test deux fois, nous obtenons alors une complétude de 1 et une résilience de $\frac{1}{4}$. En fait, vu que le Vérificateur est limité à un temps polynomial, il est possible de répéter un certain test un nombre polynomial de fois. Il s'ensuit que les trois définitions suivantes sont équivalentes.

Théorème 11 (Définitions équivalentes de IP)**.** *[Gol01] Les trois définitions suivantes de* IP *sont équivalentes :*

1. *Un langage L est dans la classe* IP *s'il existe un Vérificateur et un polynôme t tel que, pour un mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où la complétude est $\frac{2}{3}$ et la résilience $\frac{1}{3}$.*

2. *Un langage L est dans la classe* IP *si, pour tout polynôme p, il existe un Vérificateur et un polynôme t tel que, pour un mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où la complétude est $1-\frac{1}{2^{p(|x|)}}$ et la résilience est $\frac{1}{2^{p(|x|)}}$.*

3. *Un langage L est dans la classe* IP *s'il existe un Vérificateur et deux polynômes p et t tels que, pour un mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ et la différence entre la complétude et la résilience est plus grande ou égale à $\frac{1}{p(|x|)}$. De plus, la complétude et la résilience doivent être calculables en temps polynomial.*

En parallèle à la définition de IP, Babai [Bab85] a défini un autre genre de preuves interactives. En fait, il s'agit de la classe IP avec une restriction : le Vérificateur ne peut envoyer d'autres messages que des bits aléatoires. Babai cherchait à utiliser l'aléa pour pallier à notre ignorance quant à la théorie des groupes ; particulièrement en ce qui a trait au problème de l'ordre d'un groupe. Lorsqu'il a défini

cette classe, il a utilisé les noms de Arthur et Merlin plutôt que Vérificateur et Prouveur respectivement. Ces noms sont restés et permettent de différencier les deux modèles.

Définition 20 (AM[poly])**.** *Un langage L est dans la classe* AM[poly] *(Arthur-Merlin Polynomial number of messages) s'il existe un Vérificateur (Arthur) et un polynôme t tel que, pour un mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Les messages du Vérificateur sont restreints à être des chaînes aléatoires uniformément distribuées de tailles polynomiales.

Il existe une certaine confusion quant à la notation des classes du type Arthur-Merlin. À l'origine, la classe précédente était notée AM. Or, les classes qui ont le plus attiré l'attention ont été celles avec un petit nombre (constant) de messages. Rapidement, AM a désigné la classe où seuls deux messages sont transmis. Il est aussi utile de spécifier qui de Merlin ou Arthur commence le dialogue. Ainsi, la classe MA désigna la classe où il n'y a qu'un message - de Merlin à Arthur. Le A terminal signifiant qu'Arthur utilise l'aléa à l'étape finale du protocole. Il s'ensuit que la classe M est égale à la classe NP et la classe A, à la classe BPP. Nous utiliserons cette notation ainsi que la possibilité de noter AM[k] la classe définie avec k messages.

Les messages n'étant que des chaînes aléatoires, il a été possible à Babai d'obtenir des résultats intéressants tels que celui-ci :

Théorème 12. *[Bab85] Pour toute constante $k \geq 2$,*

$$\mathsf{AM}[\mathsf{k}] = \mathsf{AM}$$

.

L'obtention d'un tel résultat, qui semblait improbable à obtenir dans le paradigme de IP, paraissait indiquer que AM[poly] était strictement inclus dans IP. Aussi, comment faire quelque chose d'aussi simple que le test à l'aveugle lorsque l'on doit révéler ses bits aléatoires ? Imaginez qu'en tendant le détergent on ait à dire au représentant lequel on lui donne ! Et pourtant, les deux classes sont les mêmes !

Théorème 13. *[GS86]* $\mathsf{IP} = \mathsf{AM}[poly]$.

Nous ne donnerons pas ici la preuve mais un simple aperçu du protocole. Disons simplement que les messages du Vérificateur sont des descriptions de plusieurs fonctions de hachage. Le Prouveur doit alors donner la description d'une conversation, une étape à la fois, telle que, à chaque étape, la valeur de hachage de celle-ci est l'une des valeurs précédemment spécifiées par le Vérificateur. Si le mot est dans le langage, puisque le Prouveur doit alors avoir de bonnes chances de convaincre le Vérificateur, il se trouve que la probabilité qu'une telle conversation existe est grande et donc que le Prouveur pourra remplir une telle condition.

Les deux modèles étant équivalents, il restait à connaître la puissance même de cette classe. Il est relativement simple de montrer que $\mathsf{IP} \subseteq \mathsf{PSPACE}$. Pour ce qui est de l'inclusion inverse, une nouvelle méthode a été introduite [LFKN92] et, une semaine plus tard, Shamir [Sha92] l'utilisait pour montrer le résultat suivant :

Théorème 14. $\mathsf{IP} = \mathsf{PSPACE}$.

2.2 Quantique

Les preuves interactives quantiques ont été introduites par John Watrous en 1999 [Wat99]. Comme les autres classes de complexité quantiques déjà présentées, il s'agit d'une *quantisation* de la classe IP. Bien sûr, chacun des participants est décrit comme un circuit quantique plutôt qu'une machine de Turing. Chaque fois que deux participants échangent un message, ce sont des fils qui vont d'un circuit à l'autre. La taille du message se mesure en nombre de qubits (et donc de fils). La taille du Vérificateur est la somme des tailles des différents circuits pour les différents messages de celui-ci. Une seule mesure est effectuée, celle-ci à la toute fin du circuit, sur un seul qubit. Suit, à la figure 2.3, l'image présentée originellement dans cet article, elle représente une preuve interactive à 4 messages.

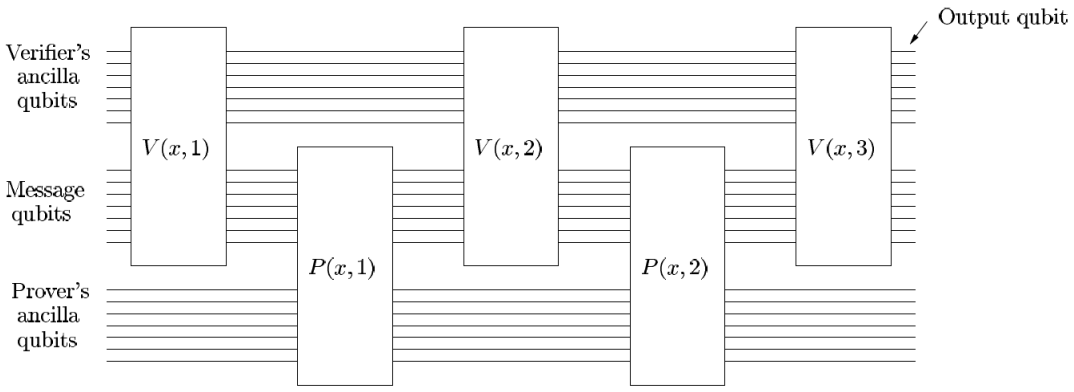

FIGURE 2.3 – Diagramme original de [Wat99]

Définition 21 (QIP)**.** *Un langage L est dans la classe* QIP *(Quantum Interactive Polynomial-Time) s'il existe un Vérificateur et un polynôme t tel que, pour un mot considéré x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Cet article [Wat99] présentait le résultat suivant : $\mathsf{IP} \subseteq \mathsf{QIP}(3)$. Un peu plus tard il fut généralisé ainsi :

Théorème 15. *[KW00]* $\mathsf{QIP} = \mathsf{QIP}(3)$.

Ceci contraste avec IP que l'on ne croit pas pouvoir être parallélisée à un nombre constant de messages. Tout récemment [JJUW09] cette classe a été caractérisée. Nous ne présentons pas la preuve ici. Qu'il suffise de dire qu'elle est basée sur le *semidefinite programming*.

Théorème 16. $\mathsf{PSPACE} = \mathsf{QIP}$.

Puisque trois messages sont suffisants, il devient tout naturel de s'intéresser tout autant à $\mathsf{QIP}(1)$ et $\mathsf{QIP}(2)$ qu'à $\mathsf{QIP}(3)$. La première inclusion du théorème suivant est triviale, pour ce qui est de la seconde, elle a été démontrée dans [MW05].

Théorème 17. $\mathsf{BQP} \subseteq \mathsf{QIP}(1) \subseteq \mathsf{PP}$.

Jusqu'à tout récemment, très peu était connu à propos de $\mathsf{QIP}(2)$ sinon les inclusions triviales $\mathsf{QIP}(1) \subseteq \mathsf{QIP}(2) \subseteq \mathsf{QIP}(3)$. Mais, étant donné le théorème 16, on peut considérer que la borne supérieure s'est considérablement améliorée, de sorte que :

Théorème 18. $\mathsf{QIP}(1) \subseteq \mathsf{QIP}(2) \subseteq \mathsf{PSPACE}$.

Tout comme QIP découle de IP, la classe $\mathsf{QAM}[\mathsf{poly}]$ peut être dérivée de la classe $\mathsf{AM}[\mathsf{poly}]$. Il en est bien sûr de même pour des classes telles que QMA et QMAM qui sont dérivées de MA et MAM respectivement. On restreint la communication du Vérificateur vers le Prouveur à être des chaînes de bits aléatoires uniformément distribuées. La notation sera elle aussi la même.

Définition 22 (QAM[poly])**.** *Un langage L est dans la classe* QAM[poly] *(Arthur-Merlin Polynomial number of messages) s'il existe un Vérificateur (Arthur) et un polynôme t tel que, pour un mot considéré x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Les messages du Vérificateur sont restreints à être des chaînes aléatoires uniformément distribuées.

La restriction qui termine cette définition semble être encore plus importante que dans le cas où l'on dérivait AM[poly] de IP puisque cette fois, alors que dans un cas les messages sont quantiques, dans l'autre, ils ne peuvent qu'être classiques. Pourtant, encore une fois, ces deux classes sont équivalentes

Théorème 19. *[MW05]* $\mathsf{QMAM} = \mathsf{QIP}(3)$.

La classe la plus connue et probablement la plus étudiée est QMA. Mentionnons que, par définition, $\mathsf{QIP}(1) = \mathsf{QMA}$. Mais surtout, QMA est considérée comme la version quantique de NP. Il s'agit donc de la classe des assertions que l'on peut vérifier efficacement quantiquement.

Pour ce qui est de la classe QAM, il n'est pas démontré qu'elle soit équivalente à $\mathsf{QIP}(2)$ bien qu'évidemment $\mathsf{QAM} \subseteq \mathsf{QIP}(2)$. Une borne supérieure classique a toute fois été montrée dans [MW05]. Ainsi nous pouvons situer QAM.

Théorème 20. $\mathsf{QIP}(1) \subseteq \mathsf{QAM} \subseteq \mathsf{BP} \cdot \mathsf{PP}$.

2.3 Questions

La première question concerne IP elle-même. Nous avons vu que le nombre de messages semblait influencer la taille de la classe et qu'un des résultats les plus puissants est justement que, quantiquement, il y a une limite à ce fait. La question demeure donc de savoir si la même chose s'avère classiquement. Nous considérons alors la classe IP[k] qui, tout comme AM[k], réfère à la classe IP restreinte à k messages.

Question 5. *Existe-t-il une constante k telle que* $\mathsf{IP} = \mathsf{IP}[\mathsf{k}]$ *?*

Il serait très étonnant qu'un tel k existe. On dit alors que la hiérarchie de IP ne s'effondre pas. C'est que cet effondrement aurait nombre d'effets pour différentes classes et donc, un peu comme un argument par le nombre, on en conclut qu'un tel effondrement est improbable. En particulier, la hiérarchie polynomiale s'effondrerait à son second niveau. Le lecteur intéressé pourra débuter sa lecture par [BHZ87, Sto77].

La seconde question revient encore une fois à comparer l'informatique classique à l'informatique quantique. Obnubilés par la possibilité de *paralléliser* le calcul en utilisant la superposition que permet la mécanique quantique, beaucoup se demandent s'il ne serait pas possible de faire en un seul message ce que l'on fait en deux messages classiquement.

Question 6. QMA *vs* AM.

Intuitivement, il s'agirait pour le Prouveur quantique de donner au Vérificateur une superposition des questions possibles du Vérificateur avec les réponses concaténées. Il est clair que, si le Prouveur est honnête et que la superposition contient toutes les chaînes aléatoires possibles pour ce qui est du message du Vérificateur, alors la construction fonctionne. Malheureusement, aucune preuve n'a été présentée à ce jour, quoique nombre de personnes croit que $\mathsf{AM} \subseteq \mathsf{QMA}$.

Finalement, nous tenons à mentionner une question qui a été résolue.

Question 7. $\mathsf{QIP}(2)$ *vs* PSPACE

Nous avons déjà mentionné que $\mathsf{QIP}(2) \subseteq \mathsf{PSPACE}$, démontré tout récemment [JJUW09]. Mais cette question était ouverte et d'importance depuis la définition de QIP au tournant du millénaire. Si l'on traçait alors un graphe d'inclusion des classes déjà mentionnées, la classe $\mathsf{QIP}(2)$ apparaissait comme un furoncle ! Nous-même avons travaillé sur la question tentant, de démontrer l'inclusion inverse. Même si le résultat [JJUW09] ne rend pas l'inclusion $\mathsf{PSPACE} \subseteq \mathsf{QIP}(2)$ impossible, elle rend sa preuve moins accessible en apparence puisque ceci impliquerait une caractérisation complète.

CHAPITRE 3

PLUS ON EST DE FOUS...

Nous avons précédemment présenté l'intuition derrière les preuves interactives de la manière suivante : supposons que vous n'ayez pas la capacité de prouver une assertion, il vous est possible d'interagir avec quelqu'un qui connaît déjà la preuve de façon à ce qu'il puisse vous convaincre de sa véracité. Mais se pourrait-il que même les explications fournies pas cette sommité en la matière soit au-delà de votre compréhension ? Bien sûr. Est-ce que tout est perdu ? Bien sûr que non. Il vous est possible d'interagir avec plus d'une sommité et de voir si leurs discours concordent. C'est une situation courante dans le quotidien qu'il ne faut pourtant pas confondre avec cette autre : souvent, on oppose des sommités, par exemple en politique, et, basé sur le débat, on se fait notre opinion. Au contraire, dans ce qui suit, toutes les sommités sont du même avis. C'est bien ça, chaque Prouveur veut vous convaincre de l'appartenance d'un mot à un certain langage.

Par exemple, vous ne connaissez rien à la mécanique et votre mécanicien vous dit qu'il faut faire impérativement une réparation pour 2000$. Que faites-vous ? Vous allez voir un autre garagiste et, sans lui dire ce que le premier vous a dit, vous lui demandez un diagnostic. S'il vous dit quelque chose de semblable au premier, vous penserez qu'ils disent vrai. Sinon, vous penserez qu'au moins l'un des deux ment. Ce qu'il faut retenir de l'exemple est que l'on peut utiliser l'interaction avec un Prouveur pour vérifier ce qu'un autre Prouveur dit. Il est bien sûr nécessaire que les deux Prouveur ne communiquent pas entre eux sinon il ne s'agit plus que d'un seul Prouveur. En effet, ils peuvent se communiquer l'un l'autre les questions et s'entendre sur les réponses à fournir au fur et à mesure du protocole.

3.1 Classique

Tel que dit précédemment, nous considérons ici un paradigme de preuves interactives où un Vérificateur interagit avec plusieurs Prouveur. Tout comme pour IP, nous considérons chacun des participants comme étant une machine de Turing et le modèle d'interaction entre le Vérificateur et chacun des Prouveur est le même. Ce modèle a été introduit [BGKW88] afin de proposer une alternative aux fonctions à sens unique pour la réalisation pratique des preuves à divulgation nulle (ZK).

Bien sûr, puisqu'il y a ici plusieurs Prouveur, il faut spécifier leur nombre. En particulier, la complétude et la résilience ne doivent plus être pensées en terme d'un seul Prouveur mais d'un ensemble de Prouveur. En fait, dans la définition originale, nous considérons un nombre polynomial de Prouveur ; il peut donc y avoir un Prouveur différent pour chacun des messages.

Définition 23 (MIP)**.** *Un langage L est dans la classe* MIP *(Multi-Prover Interactive Polynomial-Time) s'il existe un Vérificateur et un polynôme t tels que, pour tout mot x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Le Vérificateur peut interagir avec un nombre arbitraire de Prouveur.

Déjà, la question est de savoir si le nombre de Prouveur importe. Bien sûr, on s'attend à ce que deux Prouveur soient plus utiles qu'un seul. Mais de la même façon, un troisième pourrait-il nous aider à mieux vérifier ce que les deux premiers ont dit ? Cette logique s'étend-elle jusqu'à un nombre polynomial de Prouveur ou arrive-t-il un moment où ajouter des Prouveur n'aide plus ? Mentionnons aussi cette question, réminiscente de IP, à savoir si un nombre constant de messages est suffisant pour tout langage dans MIP. Nous voyons donc qu'il y a deux paramètres à spécifier lorsque l'on considérera MIP.

Définition 24 ($\mathsf{MIP}[\mathsf{a},\mathsf{b}]$)**.** *Un langage L est dans la classe* $\mathsf{MIP}[\mathsf{a},\mathsf{b}]$ *si le langage est dans la classe* MIP *où le Vérificateur n'interagit qu'avec* a *Prouveur et l'interaction avec chacun des Prouveur est limité à* 2b *messages.*

Par exemple, la classe MIP pourrait être notée $\mathsf{MIP}[\mathsf{poly},\mathsf{poly}]$. Pour des raisons historiques, l'interaction entre le Vérificateur et un Prouveur est quantifiée en *round* (un message de la part de chacun des participant) plutôt qu'en nombre de messages. Il serait plus logique, et précis, de mentionner le nombre de messages. Mais, afin que le lecteur puisse facilement faire le lien entre les classes ici mentionnées et la notation utilisée dans la littérature, nous avons choisi d'utiliser cette première manière de faire. Par contre, la majorité du temps, le lecteur verra MIP comme désignant *Multi-Prover Interactive Proof*. Puisque IP signifie *Interactive Polynomial-time*, il est plus cohérent de nommer MIP comme nous l'avons fait : *Multi-Prover Interactive Polynomial-Time*. Un autre nom serait une concession à la coutume que nous ne ferons pas.

Toujours dans ce même article [BGKW88], il a été montré que deux Prouveur sont en fait suffisants. C'est à dire que $\mathsf{MIP} = \mathsf{MIP}[2,\mathsf{poly}]$. L'idée de la preuve est de demander au Prouveur P_1 de prendre la place de tous les Prouveur dans le protocole original. Puis, pour chaque *round*, le Vérificateur choisit au hasard un des Prouveur du protocole original et demande au Prouveur P_2 de le simuler. Si les conversations avec P_1 et P_2 correspondent pour chaque round et que l'ensemble des conversations est concluant, le Vérificateur accepte. Pour k Prouveur dans le protocole original, le Vérificateur répète ceci k fois afin d'amplifier la résilience.

Cette amplification a été l'embûche à la réduction du nombre de messages tout en conservant un nombre constant de Prouveur. Suite à quelques résultats intermédiaires [FRS88, Kil90, Fei91, LS91], le résultat suivant a été obtenu :

Théorème 21. *[FL92]* $\mathsf{MIP} = \mathsf{MIP}[2,1]$

Aussi reste-t-il à situer cette classe. Avant même de connaître le résultat précédent, la question avait été résolue. Le théorème suivant est venu dans la foulée des résultats concernant IP tels que, bien sûr, $\mathsf{IP} = \mathsf{PSPACE}$ [Sha92].

Théorème 22. *[BFL91]* $\mathsf{MIP} = \mathsf{NEXP}$.

3.2 Un peu classique, un peu quantique

La communication est l'enjeu capital dans un modèle à plusieurs Prouveur. Et, de ce fait, considérer l'aspect quantique d'un modèle vient bien avant que celui-ci soit totalement quantique. Avant de considérer un Vérificateur qui puisse être quantique, on peut considérer que les Prouveur puissent, eux, l'être. Ceci ne change rien à leur puissance de calcul, mais ils peuvent alors partager de l'intrication. On se souviendra que l'intrication est une caractéristique d'un état quantique qui ne permet pas de communiquer et pourtant permet d'effectuer certaines tâches qui autrement seraient impossible ; mentionnons par exemple la pseudo-télépathie [BBT05].

Définition 25 (MIP^*)**.** *[CHTW04] Un langage L est dans la classe MIP^* (Multi-Prover Interactive Polynomial-Time with Quantum Provers) s'il existe un Vérificateur (classique) et un polynôme t tels que, pour tout mot considéré x, le Vérificateur s'arrête après un temps $O(t(|x|))$ où :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Le Vérificateur peut interagir avec un nombre arbitraire de Prouveur. On considère que les Prouveur sont quantiques ; ceux-ci peuvent donc partager un état intriqué de taille arbitraire.

L'intrication entre les Prouveur fait en sorte qu'il est très difficile de situer la classe MIP^* par rapport aux autres classes ; comme nous le verrons dans la section des questions ouvertes, nous ne savons pas comparer MIP^* et MIP. Le mieux que nous puissions présenter comme borne inférieure est, étant donné le Vérificateur classique, PSPACE. Il suffit de considérer que le Vérificateur n'interagit qu'avec un Prouveur ; l'intrication entre les Prouveur n'est alors plus un enjeu et le paradigme est celui de IP.

Théorème 23. $\mathsf{PSPACE} \subseteq \mathsf{MIP}^*$.

L'incapacité à répondre à une question fait considérer des questions intermédiaires aux chercheurs, potentiellement plus aisées. C'est ainsi que l'on en est venu à s'attarder à la classe $\mathsf{MIP}^*[2,1]$.

Définition 26 ($\mathsf{MIP}^*[2,1]$)**.** *La classe* $\mathsf{MIP}^*[2,1]$ *Est la classe* $\mathsf{MIP}[2,1]$ *où l'on considère les Prouveur comme étant quantiques ; ceux-ci peuvent donc partager de l'intrication.*

Il est important de remarquer que, contrairement au cas purement classique, nous ne connaissons pas d'équivalence entre MIP^* et $\mathsf{MIP}^*[2,1]$. En fait, nous ne connaissons aucune relation entre ces deux classes. Tout ce qu'elles partagent est la borne inférieure. Et ce résultat n'est pas trivial à obtenir. Il a été précédé d'un autre résultat où l'on a dû considérer une complétude polynomialement proche de 1 [KKM$^+$08]. Ce n'est que dernièrement qu'une borne exponentiellement proche de 0 a été obtenue.

Théorème 24. *[IKM08]* $\mathsf{PSPACE} \subseteq \mathsf{MIP}^*[2,1]$.

Afin de bien extraire l'intuition derrière le résultat précédent, rappelons que nous ne savons pas ramener à une constante le nombre de messages dans un paradigme où il n'y a qu'un Prouveur. Le fait que nous obtenions un certain parallélisme lorsque nous considérons deux Prouveur quantiques laisse penser que ce modèle est

plus puissant que celui de IP, en dépit de l'intrication que partagent ceux-ci. Malgré tout, ce dernier théorème ne donne pas une indication claire de la puissance du paradigme à plusieurs Prouveur quantiques. Il serait préférable de comparer $\mathsf{MIP}^*[2,1]$ à une classe plus grande que PSPACE. De plus, nous savons que $\mathsf{MIP} = \mathsf{NEXP}$, il est donc naturel de vouloir comparer $\mathsf{MIP}^*[2,1]$ à NEXP. Malheureusement, pour ce faire, nécessaire de considérer une résilience qui soit proche de 1. Nous ne savons pas, en effet, obtenir une meilleur résilience face à l'intrication que partagent les Prouveur dans ce paradigme.

Théorème 25. *[IKM08]* $\mathsf{NEXP} \subseteq \mathsf{MIP}^*[2,1]$ *avec complétude 1 et résilience* $1 - \frac{1}{exp(n)}$ *où n est la taille du mot considéré.*

3.3 Quantique

Après avoir considéré que les Prouveur sont quantiques, considérons maintenant que le Vérificateur l'est lui aussi. Cette classe est peut-être plus naturelle à étudier que MIP^*. Il est peut-être plus aisé pour un Vérificateur quantique de tirer parti de l'intrication possible entre les Prouveur. Clairement, $\mathsf{MIP}^* \subseteq \mathsf{QMIP}$. Mais comme nous le verrons, nous ne savons pas beaucoup plus.

Définition 27 (QMIP)**.** *Un langage L est dans la classe* QMIP *(Quantum Multi-Prover Interactive Polynomial-Time) s'il existe un Vérificateur quantique et un polynôme t tel que, pour un mot considéré x, le Vérificateur est de taille* $O(t(|x|))$*, et :*

- *La complétude du protocole est* $\frac{2}{3}$*.*
- *La résilience du protocole est* $\frac{1}{3}$*.*

Le Vérificateur peut interagir avec un nombre arbitraire de Prouveur.

Aussi, le Vérificateur étant quantique, il est naturel d'anticiper que MIP soit strictement contenu dans QMIP. Les possibilités que permet la communication quantique laisse aussi anticiper que le paradigme de QMIP soit très puissant. Et pourtant, seule la déception attend le lecteur même si cette classe a été définie [KM03] dès 2003.

Semblablement à MIP^* vs. PSPACE, il est trivial d'établir une borne inférieure pour la classe QIP en considérant que le Vérificateur n'interagit qu'avec un Prouveur.

Théorème 26. $\mathsf{QIP} \subseteq \mathsf{QMIP}$.

On voudrait pourtant, et bien naturellement, établir une relation entre MIP et QMIP. On se souviendra que $\mathsf{MIP} = \mathsf{NEXP}$. Malheureusement, aucune relation n'a pu être établie. Encore une fois, les chercheurs se sont alors penchés sur une classe qui leur semblait plus aisée à analyser : $\mathsf{QMIP}[2,1]$. Et alors, un résultat, bien que peu concluant étant donnée la résilience, a été obtenu.

Théorème 27. *[KKM+08]* $\mathsf{NEXP} \subseteq \mathsf{QMIP}[2,1]$ *avec résilience* $1 - \frac{1}{exp(n)}$ *où* n *est la taille du mot considéré.*

Il existe une façon de simplifier le paradigme de QMIP autre que de restreindre le nombre de Prouveur. Tout comme il est plus simple d'analyser MIP que MIP^*, du fait de l'intrication possible entre les Prouveur, il est plus simple de considérer le paradigme de QMIP en y soustrayant l'intrication entre les Prouveur. Il est alors possible de montrer une caractérisation complète de la classe.

Définition 28 ($\mathsf{QMIP}_{\mathsf{ne}}$)**.** *[KM03] Un langage L est dans la classe $\mathsf{QMIP}_{\mathsf{ne}}$ (Quantum Multi-Prover Interactive Polynomial-Time With No Prior Entanglement) s'il existe un Vérificateur et un polynôme t tel que, pour tout mot considéré x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Le Vérificateur peut interagir avec un nombre arbitraire de Prouveur. On restreint les Prouveur à ne partager aucune intrication.

Théorème 28. $\mathsf{QMIP}_{\mathsf{ne}} = \mathsf{NEXP}$.

Il est étonnant de voir que le quantique s'avère inutile dans le paradigme des preuves interactives à plusieurs Prouveur lorsque ceux-ci n'ont pas la possibilité de partager de l'intrication. Le pessimisme n'est toutefois pas de rigueur. Une autre façon de percevoir ce résultat est de voir que l'interaction avec plusieurs Prouveur est à ce point puissante que la mécanique quantique devient inutile. Mais puisque nous ne savons pas si la mécanique quantique permet un réel gain d'efficacité pour le calcul...

Comme nous l'expliquerons plus en détail dans la dernière section de ce chapitre, nous ne savons pas si permettre aux Prouveur de partager de l'intrication est un gain pour le Prouveur ou pour le Vérificateur. En d'autres mots, nous ne savons pas si restreindre la quantité d'intrication partagée fait en sorte que la classe soit plus petite, équivalente, plus grande ou si les deux modèles sont incomparables. La première étape pour répondre à la question est donc d'introduire *un peu* d'intrication dans le modèle.

Définition 29 ($\mathsf{QMIP}_{\mathsf{le}}$)**.** *[KM03] Un langage L est dans la classe $\mathsf{QMIP}_{\mathsf{le}}$ (Quantum Multi-Prover Interactive Polynomial-Time with Limited Prior Entanglement) s'il existe un Vérificateur et un polynôme t tel que, pour un mot considéré x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est $\frac{2}{3}$.*
- *La résilience du protocole est $\frac{1}{3}$.*

Le Vérificateur peut interagir avec un nombre arbitraire de Prouveur. On restreint les Prouveur à ne partager qu'une quantité polynomiale $O(p(|x|))$ de qubits d'intrication.

Le résultat suivant semble indiquer que le partage d'intrication entre les Prouveur réduit la taille de la classe. Mais ceci n'est qu'une indication. Premièrement parce qu'il est possible que les deux classes soient équivalentes. Deuxièmement, il n'est pas du tout à exclure qu'une quantité plus grande d'intrication permette, contrairement à $\mathsf{QMIP}_{\mathsf{le}}$, une classe incluant NEXP.

Théorème 29. $\mathsf{QIP} \subseteq \mathsf{QMIP}_{\mathsf{le}} \subseteq \mathsf{NEXP}$.

3.4 Questions

Le présent chapitre a principalement introduit trois paradigmes de preuves interactives : le premier avec plusieurs Prouveur classiques, le second avec des Prouveur pouvant partager de l'intrication et le dernier totalement quantique. Malheureusement, nous ne savons pas bien les comparer.

Question 8. MIP *vs* MIP^* *et* MIP *vs* QMIP.

Regardons ces trois classes une à une. Nous savons déjà que $\mathsf{MIP} = \mathsf{NEXP}$. Comme dit précédemment, toute la puissance de MIP est due au fait que les deux

Prouveur ne peuvent pas communiquer entre eux durant le protocole de vérification. Il semble donc que de permettre au Prouveur de partager de l'intrication affaiblirait le modèle et donc que $\mathsf{MIP}^* \subseteq \mathsf{MIP}$. Intuitivement, on peut s'attendre à ce que les Prouveur aient alors la capacité de résoudre certains problèmes semblablement à ce qui est fait en pseudo-télépathie [BBT05]. Mais à l'opposé, on pourrait considérer que le Vérificateur puisse être plus exigeant du fait que les Prouveur aient plus de ressources. Le défi étant plus grand, la classe pourrait contenir un plus grand nombre de langages. Il n'est donc pas inconcevable que $\mathsf{MIP} \subseteq \mathsf{MIP}^*$. Nous envisageons toutefois qu'une troisième option est la bonne : le recours à l'intrication par les Prouveur exclut certains langages de MIP cependant qu'il en inclut d'autres. Il n'y aurait donc pas de relation directe entre ces deux classes. Par contre, il se pourrait bien évidemment que ces deux classes soient égales.

De façon évidente, $\mathsf{MIP}^* \subseteq \mathsf{QMIP}$; il suffit au Vérificateur d'agir tout comme s'il était classique. Le fait que le Vérificateur soit quantique pour cette dernière classe laisse penser que $\mathsf{MIP} \subset \mathsf{QMIP}$. Or, le même raisonnement concernant l'intrication s'applique. Il y a toutefois une différence fondamentale, il se pourrait qu'un Vérificateur quantique soit bien plus en mesure de capitaliser sur l'intrication entre les Prouveur. L'inclusion $\mathsf{MIP} \subseteq \mathsf{QMIP}$ est donc plus vraisemblable que celle $\mathsf{MIP} \subseteq \mathsf{MIP}^*$. Par contre, le résultat $\mathsf{QMIP}_{\mathsf{le}} \subseteq \mathsf{MIP}$ laisse plutôt entendre que l'inclusion inverse est à prévoir.

Tout ceci amène une question plus précise :

Question 9. *Quel est l'impact de l'intrication entre les Prouveur ?*

Nous avons comme début de réponse que $\mathsf{QMIP}_{\mathsf{ne}} = \mathsf{NEXP}$ et que $\mathsf{QMIP}_{\mathsf{le}} \subseteq \mathsf{NEXP}$. Ceci laisse croire que l'intrication est nuisible, du point de vue du Vérificateur. Mais se pourrait-il qu'une quantité exponentielle de qubits d'intrications entre les Prouveur change totalement la classe ? Peut-on imaginer une hiérarchie complète qui dépendrait de la quantité d'intrication entre les Prouveur ? Toutes

ces questions, si elles avaient des réponses, nous permettraient non seulement de résoudre des problèmes de complexité, mais nous mèneraient aussi à une meilleure compréhension de ce qu'est l'intrication.

Toujours dans le but de comprendre la nature de l'intrication, on en est venu à créer des classes telles que $\mathsf{MIP}^*[2,1]$ qui semblaient simplifier le problème. Mais cela amène aussi de nouvelles questions telles que :

Question 10.

MIP^* *vs* $\mathsf{MIP}^*[2,1]$.

Le lecteur se souviendra du résultat suivant : $\mathsf{MIP} = \mathsf{MIP}[2,1]$. La création de la classe $\mathsf{MIP}^*[2,1]$ a pour origine ce résultat. Et à ce jour, nous ne connaissons rien de l'influence du nombre de messages ou de Prouveur sur la classe. Il se pourrait par exemple que $\mathsf{MIP}^*[2,1]$ soit pour longtemps une classe mystérieuse comme l'a été $\mathsf{QIP}(2)$ alors qu'une classe avec plus de Prouveur ou plus de messages serait plus accessible à l'analyse. L'influence de l'intrication pourrait être la même que pour QMIP et mène aux mêmes supputations. Notons que, par contre, le Vérificateur étant classique dans un cas et quantique dans l'autre, sa capacité à capitaliser sur l'intrication des Prouveur pourrait être bien différente et mener à des résultats dissemblables.

CHAPITRE 4

TOUT EST DANS LA MANIÈRE

Nous avons débuté cette thèse en présentant la classe P, puis, toujours en considérant des machines de Turing fonctionnant en temps polynomial, nous avons présenté aux chapitres 2 et 3 une panoplie de classes interactives. Le Vérificateur étant une machine de Turing fonctionnant en temps polynomial, il était naturel de considérer qu'il reçoive des messages de taille polynomiale. Considérons maintenant la possibilité de messages de taille plus modeste.

4.1 Message de taille logarithmique

Ne considérons d'abord qu'un message, classique, et qui soit de taille logarithmique. On pourrait nommer cette classe $\mathsf{NP}_{\log}$. Un tel message peut-il être utile à un Vérificateur fonctionnant en temps polynomial ? On se souviendra qu'à la base un Prouveur est tout puissant et que, si des messages susceptibles de convaincre le Vérificateur existent, celui-ci s'attend à recevoir l'un d'eux. Mais comme ici le message est de taille logarithmique, le Vérificateur peut lui-même chercher ce message. En effet, il n'y a qu'un nombre polynomial de messages d'une telle taille et le vérificateur n'a qu'à *vérifier* chacun d'eux. Il acceptera donc le mot si et seulement si un tel message existe et pourra décider d'un langage dans cette classe sans le concours d'un Prouveur. Nous employons le terme Vérificateur pour mettre en évidence son rôle ; une fois le Prouveur évincé, le Vérificateur n'est pourtant plus qu'une machine de Turing traditionnelle.

Le fait que l'existence d'un message valide soit suffisant facilite grandement la tâche du Vérificateur dans l'exemple précédent. Qu'en est-il alors d'une classe où la décision du Vérificateur est probabiliste ? En effet, une classe telle que MA avec un message de taille logarithmique (nommons cette classe $\mathsf{MA}_{\log}$). Encore une fois, un

Prouveur serait inutile. Tout comme dans l'exemple précédent, le Vérificateur peut considérer tous les messages possibles. Cette fois par contre, il ne fera pas qu'un test sur chacun des messages mais un nombre linéaire de tests. En effet, par définition de la classe, la probabilité d'erreur (la résilience et le manque pour une complétude parfaite) est au plus $\frac{1}{3}$. Pour chaque message, le Vérificateur effectuera $\Theta(n)$ tests et acceptera si et seulement si la proportion de tests réussis est supérieure à $\frac{1}{2}$. Par la borne de Chernoff, on voit que la probabilité d'erreur pour un message est alors bornée par $e^{-\Omega(n)}$ pour n vérifications sur un message. Comme le nombre de messages est polynomial, le Vérificateur peut alors évaluer s'il existe un message tel que la probabilité d'acceptation est au moins $\frac{2}{3}$ et ce avec une probabilité d'erreur exponentiellement petite. S'il existe un tel message, le Vérificateur accepte le mot. Autrement, il le refuse. Tout ceci encore une fois sans l'assistance d'un Prouveur.

Dans les deux exemples précédents, le Vérificateur avait la possibilité de considérer un à un tous les messages possibles. Il pouvait alors gérer le côté probabiliste de la classe. Mais qu'en est-il dans le cas d'un continuum de messages ? Cette question réfère bien entendu aux preuves quantiques (de taille logarithmique) ; le message du Prouveur est-il nécessaire au Vérificateur ? Étonnamment non, et, encore plus étonnant, celui-ci peut vérifier tous les messages possibles, d'une certaine façon. Ce résultat a été présenté dans [MW05] et nous l'exposons ici. Nous débutons en présentant ce qu'est une classe telle QMA mais avec des messages de taille variable.

Définition 30 (QMA_m[c, r])**.** *Un langage L est dans la classe $\mathsf{QMA}_\mathsf{m}[c, r]$ (Quantum Arthur-Merlin message of size m) s'il existe un Vérificateur quantique et un polynôme t tel que, pour tout mot x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est $c(|x|)$.*
- *La résilience du protocole est $r(|x|)$.*

Il n'y a qu'un seul message, du Prouveur au Vérificateur, de taille $O(m)$.

Ici, nous nous intéresserons à la classe $\mathsf{QMA}_{\log} = \mathsf{QMA}_{\log}[\frac{2}{3}, \frac{1}{3}]$. Dans le même article [MW05], il a été montré que pour tout message de taille au plus polynomiale, il est possible d'atteindre une erreur exponentiellement faible et ce sans augmenter la taille du message. C'est ce que formalise le lemme suivant.

Lemme 2 (Lemme d'amplification)**.** *[MW05] Soit* $\mathsf{QMA}_{\mathsf{m}}[c, r]$. *Soit* p *un polynôme alors si* $c(n)$ *et* $r(n)$ *sont tels que*

$$c(n) - r(n) \geq \frac{1}{p(n)}$$

pour tout n *alors*

$$\mathsf{QMA}_{\mathsf{m}}[c(n), r(n)] = \mathsf{QMA}_{\mathsf{m}}[1 - 2^{-q}, 2^{-q}]$$

pour tout q *et* m *dans* $O(poly(n))$.

Avoir une erreur exponentiellement petite nous permettra de décrire comment le Vérificateur peut vérifier un langage de la classe $\mathsf{QMA}_{\log}$, sans le message du Prouveur, et ceci avec une probabilité d'erreur constante. Mais d'abord, nous aurons besoin du lemme suivant. Nous présentons celui-ci aussi conformément que possible à ce qui a été présenté dans l'article original.

Lemme 3. *[MW05] Soit un Vérificateur qui (sans perte de généralité), lorsqu'il interagit avec le Prouveur, a le comportement suivant : étant donné le mot* x*, le Vérificateur appliquera une certaine transformation unitaire* (A_x) *sur l'ensemble du registre de messages (de taille* m*) et celui de ses qubits ancillaires (de taille* k*). Puis il fera une mesure dans la base de calcul sur le premier qubit. Nous noterons* Π_1 *le projecteur correspondant à obtenir 1 sur cette mesure. Soit* Q_x*, une matrice* $m \times m$*, comme suit :*

$$Q_x = (I_m \otimes \langle 0^k|) A_x^\dagger \Pi_1 A_x (I_m \otimes |0^k\rangle)$$

Si le Vérificateur a une probabilité d'erreur exponentiellement petite alors $\forall L \in \mathsf{QMA}_{\log}$,

1. *Si* $x \in L$ *alors* $Tr(Q_x) \geq \frac{2}{3}$.

2. *Si* $x \notin L$ *alors* $Tr(Q_x) \leq \frac{1}{3}$.

Démonstration. Q_x est, par définition, une matrice positive et la probabilité d'acceptation du message $|\psi_x\rangle$ est $\langle\psi_x|Q_x|\psi_x\rangle$. En effet, il est aisé de s'en rendre compte en décomposant comme suit $\langle\psi_x|Q_x|\psi_x\rangle$. D'une part, $A_x^\dagger \Pi_1 A_x$ est lui-même un projecteur de taille $mk \times mk$ (que nous nommerons P_x). D'autre part, $(I_m \otimes |0^k\rangle)|\psi_x\rangle = |\psi_x\rangle|0^k\rangle$, un vecteur de taille mk. Nous avons donc que la probabilité d'avoir un résultat de mesure correspondant au projecteur P_x, étant donné l'état $|\psi_x\rangle|0^k\rangle$, est $\langle\psi_x|\langle 0^k|P_x|\psi_x\rangle|0^k\rangle$.

La probabilité maximale d'acceptation du Vérificateur est égale à la plus grande valeur propre de Q_x. Par définition, la probabilité d'erreur peut être réduite à 2^{-m-2}. La trace d'une matrice est égale à la somme de ses valeurs propres. Soulignons aussi que toutes les valeurs propres de Q_x sont positives. Donc, d'une part, nous savons que, si $x \in L$, l'une des valeurs propres est au moins $1-2^{-m-2}$. D'autres part, si $x \notin L$, nous savons que les 2^m valeurs propres de la matrice sont au plus 2^{-m-2}. Nous en arrivons donc à ce que :

1. Si $x \in L$ alors $Tr(Q_x) \geq 1 - 2^{-m-2} \geq \frac{2}{3}$.

2. Si $x \notin L$ alors $Tr(Q_x) \leq 2^m 2^{-m-2} \leq \frac{1}{3}$.

□

Avec ce lemme en main, nous pouvons maintenant décrire comment le Vérificateur peut simuler le message du Prouveur.

Théorème 30. $\mathsf{QMA}_{\log} = \mathsf{BQP}$.

Démonstration. Trivialement, $\mathsf{BQP} \subseteq \mathsf{QMA}_{\log}$. Nous montrerons ici que $\mathsf{QMA}_{\log} \subseteq \mathsf{BQP}$.

Semblablement au cas classique, le Vérificateur voudrait essayer sa procédure de vérification sur tous les messages possibles. Dans le cas présent, il construira un état qui, c'une certaine manière, contiendra le message qu'il recevrait du Prouveur. En effet, en partant de ce message ($|\psi_x\rangle$), il est possible de construire une base orthonormale contenant ce vecteur. La superposition de ces états correspond à l'état parfaitement mélangé. Et il peut, bien sûr, construire cet état sans connaître $|\psi_x\rangle$. Le Vérificateur créera donc un état complètement mélangé ($2^{-m}I_m$) et utilisera celui-ci comme s'il s'agissait du message du Prouveur. La probabilité pour le Vérificateur original d'accepter un message $|\psi_x\rangle$ est, tel que présenté dans la preuve du théorème précédent, $\langle\psi_x|Q_x|\psi_x\rangle = Tr(Q_x|\psi_x\rangle\langle\psi_x|)$. La probabilité d'accepter ce message artificiel est donc :

$$Tr(Q_x 2^{-m} I_m) = 2^{-m} Tr(Q_x)$$

Mais on se rappellera que $m \in O(\log |x|)$. Il en résulte que la différence entre la complétude et la résilience est $\Omega(|x|)^{-1}$. Il est donc aisé d'amplifier celle-ci par des méthodes standards de répétitions. Nous pouvons en conclure que $\mathsf{QMA}_{\log} \subseteq \mathsf{BQP}$. □

4.2 Travaux antérieurs

De la section précédente, il semble que l'on puisse conclure à l'inutilité d'une preuve interactive de taille logarithmique pour un Vérificateur fonctionnant en temps polynomial. Il est pourtant possible d'aller un peu plus loin. Supposons que l'on donne deux preuves à feuilleter au Vérificateur, qu'advient-il ? On peut en effet considérer un modèle où le Prouveur fournit deux preuves au Vérificateur.

Quelle est la différence entre recevoir une preuve et deux preuves ? Le modèle que nous abordons maintenant renferme, dans sa définition, la promesse que les deux registres ne sont pas intriqués, c'est ainsi que l'on peut compter le nombre de preuves.

Le fait d'avoir deux preuves permet, tout comme dans les modèles à plusieurs Prouveur, de contre-vérifier certaines caractéristiques de celles-ci. Cette absence d'intrication semble essentielle et, comme nous le verrons, permet de définir une classe de preuves interactives de taille logarithmique qui contienne BQP. Notons, par ailleurs, que le Vérificateur ne peut utiliser la même technique d'amplification que s'il avait une seule preuve. En effet, l'état complètement mélangé pris sur chacun des registres correspond à l'état complètement mélangé sur l'ensemble du message. Ceci n'exclut donc pas le cas où les deux registres sont intriqués ; la preuve qui convainc le Vérificateur pourrait donc ne pas être conforme à la définition de la classe.

La classe précédemment décrite a été définie dans [BT09a] où il a été montré que tout langage dans NP est inclus dans cette classe avec une complétude parfaite et une résilience polynomialement proche de 1. Voici la formalisation de la définition et du résultat.

Définition 31 ($\mathsf{QMA}_{\log}(2)$)**.** *Un langage L est dans la classe $\mathsf{QMA}_{\log}(2)$ (Quantum Merlin-Arthur two logarithmic size messages) s'il existe un Vérificateur quantique et un polynôme t tel que, pour tout mot x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est* 1.
- *La résilience du protocole est* $1 - \frac{1}{\mathsf{poly}(n)}$.

Il n'y a qu'un seul message, du Prouveur au Vérificateur. Ce message est composé de deux registres non-intriqués de taille $O(\log(t(|x|)))$.

Théorème 31. $\mathsf{NP} \subseteq \mathsf{QMA}_{\log}(2)$.

La preuve de ce théorème est semblable à celle du théorème 34, nous l'omettons donc.

Tel que dit précédemment, nous ne savons pas amplifier la résilience pour cette classe. Il semble que deux registres ne soient pas suffisants pour atteindre une résilience constante. Indépendamment de nos travaux, une classe comportant plus de deux registres a été étudiée [ABD$^+$08]. Cet article comporte la preuve qu'il est possible d'atteindre des complétude et résilience constantes en utilisant un nombre de registres non-intriqués inférieur à un nombre polynomial.

Définition 32 ($\mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$)**.** *Un langage L est dans la classe $\mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$ (Quantum Merlin-Arthur $\sqrt{n}\mathsf{polylog}(n)$ logarithmic size messages) s'il existe un Vérificateur quantique et un polynôme t tel que, pour tout mot x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est* 1.
- *La résilience du protocole est* $\frac{1}{3}$.

Il n'y a qu'un seul message, du Prouveur au Vérificateur. Ce message est composé de $\sqrt{n}\mathsf{polylog}(n)$ registres non-intriqués de taille $O(\log(t(|x|)))$.

Il y a une précision importante à apporter lorsque l'on spécifie que le nombre de registres est inférieur à un nombre polynomial. La preuve a été faite pour le langage $3\mathcal{SAT}$. Ce langage est NP-complet selon la réduction polynomiale. Il s'ensuit que le protocole présenté, pour un autre langage dans NP, pourrait nécessiter un nombre polynomial de registres non-intriqués. Le résultat présenté dans [ABD$^+$08] ne permet donc pas de conclure que $\mathsf{NP} \subseteq \mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$ mais seulement que :

Théorème 32. $3\mathcal{SAT} \subseteq \mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$

Dans l'obtention de ce résultat, les auteurs ont commenté nos travaux préliminaires [BT07], mentionnant qu'il serait étonnant que deux messages soient suffisants pour obtenir une résilience constante puisque ceci pourrait potentiellement mener à l'égalité $\mathsf{QMA}(2) = \mathsf{NEXP}$. En effet, cette égalité est semblable à $\mathsf{QMA}_{\log}(2) = \mathsf{NP}$. La différence majeure est que dans la première égalité, un Vérificateur devra faire face, en temps polynomial, à des registres de tailles polynomiales plutôt que logarithmique comme dans la seconde égalité.

Ces travaux ([ABD+08]), ainsi que les nôtres, permettent de mettre en relief le comportement de l'intrication. L'absence d'intrication semble nuire au Prouveur, lui qui avait toute la capacité d'en tirer partie. Au contraire, le Vérificateur peut, lui, tirer partie de cette absence qui lui est promise. Aussi, un nombre $\sqrt{n}\mathsf{polylog}(n)$ de registres non-intriqués permet d'atteindre une résilience constante alors qu'un nombre constant de ceux-ci ne semble pas suffire. Mais au-delà de ça, ces deux résultats sont une étape intermédiaire dans le dessein de rapprocher NP de BQP. En effet, ces deux classes n'ont pas de relations connues. Les classes $\mathsf{QMA}_{\log}(2)$ et $\mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$ ont la caractéristique d'inclure BQP de même que des problèmes NP-complets. En fait, on peut présenter les présenter, dans l'état actuel de nos connaissances, comme la plus petite aide requise (la preuve du Prouveur) par une machine quantique pour pouvoir vérifier des problèmes dans NP en temps polynomial.

4.3 Caractérisation de NP

L'interprétation précédente de la classe $\mathsf{QMA}_{\log}(2)$ mène naturellement à la question suivante : et si l'on ne se souciait pas d'inclure BQP, à quel point pourrait-on affaiblir le modèle tout en incluant encore NP ? On veut donc une classe qui en soit la plus *proche* possible. Même si cette métrique est très floue, il est facile de

qualifier l'obtention de la réponse optimale : une caractérisation de NP. C'est à dire définir une classe égale à NP. C'est justement ce que nous avons obtenu [BT09b] et présentons ici.

Face aux classes de complexité qui existent et à toutes leurs relations connues, le but ultime est toujours de montrer l'égalité ou l'inégalité de celles-ci. Pour les classes fondamentales, ce sont les résultats les plus difficiles à obtenir et bien sûr les plus rares. Mais ce sont aussi les plus instructifs. Et plus les deux paradigmes comparés sont dissemblables, plus le résultat est instructif. Ce que nous présentons ici nous apprend donc comment l'on peut comparer le non-déterminisme avec le calcul quantique ; il ne peut y avoir plus dissemblable. Aussi, le résultat est d'autant plus intéressant lorsqu'il concerne une classe de complexité fondamentale, une classe pivot. Le présent résultat concerne la classe maîtresse s'il en est une : NP.

Puisque nous présentons la classe $\mathsf{PQMA}_{\log}(2)$ comme égale à NP, nous pouvons la présenter comme un équilibre. C'est à dire que, d'une part, il s'agit de fournir au Vérificateur suffisamment de puissance de calcul pour inclure NP. Mais d'autre part, celui-ci doit être le plus faible possible, suffisamment du moins pour être simulé par un Vérificateur classique en temps polynomial. C'est ici que toute la difficulté réside. Si, de manière générale, on s'attend à ce que la mécanique quantique nous offre des avantages en regard de la capacité de calcul, il est aisé de concevoir la difficulté que représente la simulation de celle-ci.

La définition suivante présente donc un Vérificateur comme un circuit quantique de taille polynomiale fonctionnant sur un registre de taille logarithmique. Ce circuit est créé par une machine de Turing fonctionnant en temps polynomial, la réponse donnée par le Vérificateur est le résultat de la mesure d'un qubit du circuit quantique. Il n'y a donc pas de traitement après l'exécution du circuit. Comme nous le verrons, tout ceci peut être simulé par un Vérificateur classique fonctionnant en temps polynomial. Ce Vérificateur reste toutefois suffisamment puissant pour vérifier tout langage dans NP.

Définition 33 ($\mathsf{PQMA}_{\log}(2)$)**.** *Un langage L est dans la classe $\mathsf{PQMA}_{\log}(2)$ (Polynomial-Time Quantum Arthur-Merlin two logarithmic size messages) s'il existe deux polynôme v, t et une machine de Turing classique pouvant décrire, en temps $O(v(|x|))$, un Vérificateur quantique tel que, pour tout mot considéré x, le Vérificateur est de taille $O(t(|x|))$, et :*

- *La complétude du protocole est* 1.
- *La résilience du protocole est* $1 - \frac{1}{\mathsf{poly}(n)}$.

Il n'y a qu'un message, du Prouveur au Vérificateur. Ce message est composé de deux registres non-intriqués de taille $O(\log(t(|x|)))$. Le registre de travail du Vérificateur généré est lui-même de taille $O(\log(t(|x|)))$.

Le lecteur remarquera que seul le circuit quantique est nommé Vérificateur et non pas la paire circuit-machine de Turing. Ceci dans un but d'homogénéité avec les autres définitions. Ceci met aussi l'emphase sur le rôle du circuit quantique. Par choix, nous avons défini le Vérificateur comme un circuit quantique créé en temps polynomial. La machine de Turing effectue d'abord un prétraitement. Soulignons au passage que le fait que la machine de Turing génère le circuit quantique va au-delà de l'uniformité : la machine de Turing peut effectuer des calculs et le circuit peut ne pas dépendre seulement de la taille de l'entrée. Le Vérificateur, lui, effectue naturellement son travail : il traite le message du Prouveur et l'acceptation ou le rejet du mot correspond au résultat de la mesure effectuée sur le premier qubit du registre. D'autres définitions auraient pu être données et conduire au même résultat. Mentionnons le cas ou un circuit quantique est d'abord utilisé par une machine de Turing qui, ensuite, fait un post-traitement avant de produire son verdict.

La classe ainsi définie nous permettra de conclure, suite aux théorèmes 34 et 37, que :

Théorème 33. $\mathsf{PQMA}_{\log}(2) = \mathsf{NP}$.

4.3.1 Preuve quantique de taille logarithmique

Nous montrerons d'abord l'inclusion suivante : $\mathsf{NP} \subseteq \mathsf{PQMA}_{\log}(2)$. Pour ce faire, nous expliciterons un protocole qui puisse être effectué par un Vérificateur généré par une machine de Turing tel que décrit dans la définition afin de décider du langage NP-complet $3\mathcal{COL}$.

Définition 34. *$3\mathcal{COL}$ est l'ensemble des graphes non orientés $G = (V, E)$ (utilisant un encodage naturel dans des chaînes de caractères) pour lesquels il existe un coloriage $C : V \to \{0, 1, 2\}$ tel que $\forall (x, y) \in E$, $x \neq y \Rightarrow C(x) \neq C(y)$.*

En démontrant que $3\mathcal{COL} \in \mathsf{PQMA}_{\log}(2)$ nous aurons démontré que $\mathsf{NP} \subseteq \mathsf{PQMA}_{\log}(2)$; ceci parce que la résilience est polynomialement proche de 1, la taille de chaque registre est logarithmique et leur nombre constant. Ces caractéristiques sont essentielles puisque nous considérons $3\mathcal{COL}$ comme complet étant donnée une réduction en temps polynomial. Si, par exemple, le nombre de registres avait été $\sqrt{n}$, une réduction en temps $O(n^4)$ pourrait engendrer un nombre de registres qui ne soit pas conforme avec la définition de la classe. Nous présentons maintenant le protocole en même temps que l'analyse de la complétude. Nous effectuerons ensuite la preuve concernant la résilience. Nous pourrons alors conclure au théorème suivant :

Théorème 34. $\mathsf{NP} \subseteq \mathsf{PQMA}_{\log}(2)$.

Mais nous aurons d'abord besoin des deux définitions suivantes :

Définition 35 (Swap-Test)**.** *[BCWW01] Pour deux états purs $|\Psi\rangle$ et $|\Phi\rangle$, le Swap-Test est le circuit de la figure 4.1 où la mesure est effectuée dans la base de calcul. L'opération SWAP correspond à $|\psi\rangle|\phi\rangle \mapsto |\phi\rangle|\psi\rangle$.*

Lemme 4 (Swap-Test)**.** *La probabilité de succès du Swap-Test (la probabilité de mesurer 0), étant donnés deux états purs $|\Psi\rangle$ et $|\Phi\rangle$, est $\frac{1}{2} + \frac{|\langle\Psi|\Phi\rangle|^2}{2}$*

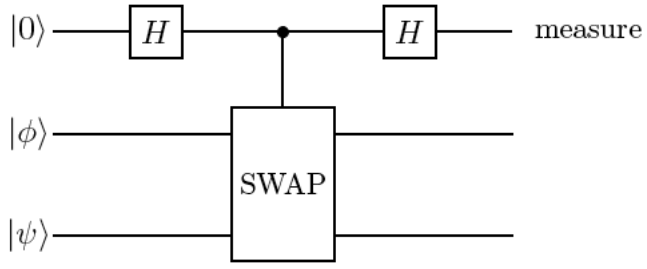

FIGURE 4.1 – Diagramme original de [BCWW01]

Définition 36 (Transformée de Fourier)**.** *La transformée de Fourier, appliquée à un état de dimension N , est définie comme suit :*

$$F_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi ixy/N}|y\rangle$$

.

Nous notons $|\Psi\rangle$ et $|\Phi\rangle$ les deux registres non intriqués constituant le message du Prouveur. Considérant un graphe de n noeuds, chacun de ces registres est dans l'espace $\mathcal{H}_n \otimes \mathcal{H}_3$ qui correspond respectivement à la partie *indice* (espace de dimension n) et à celle *couleur* (espace de dimension 3). Le Vérificateur effectuera avec probabilité $\frac{1}{3}$ l'un des tests suivants et acceptera le mot si et seulement si ce test est concluant. Mentionnons immédiatement que le certificat d'un prouveur honnête correspondra à la superposition de tous les noeuds avec leur couleur respective pour un coloriage donné.

- **Test 1 :** (Égalité des deux registres) Le vérificateur effectue le *swap-test* sur les registres $|\Psi\rangle$ et $|\Phi\rangle$ et accepte si et seulement si le test réussi.

- **Test 2 :** (Validité du coloriage) Les registres $|\Psi\rangle$ et $|\Phi\rangle$ sont mesurés dans la base de calcul. Étant donné les résultats de la mesure $(i, C(i)), (i', C'(i))$,

a) Si $i = i'$, accepter si et seulement si $C(i) = C'(i)$.

b) Sinon, si $(i, i') \in E$ accepte si et seulement si $i \neq i' \Rightarrow C(i) \neq C'(i)$.

c) Sinon, si $(i, i') \notin E$ accepter.

- **Test 3 :** (Totalité du coloriage) Indépendamment, pour chacun des registres $|\Psi\rangle$ et $|\Phi\rangle$, le Vérificateur mesure séparément les parties *indice* et *couleur* dans la base de Fourier. Le Vérificateur rejette si et seulement si le résultat de la mesure correspond à $F_3|0\rangle$ pour la partie *couleur* et que le résultat de la mesure de la partie *indice* ne correspond pas à $F_n|0\rangle$.

Ce protocole a complétude 1.

Théorème 35. *Si $x \in 3\mathcal{COL}$, alors il existe un message quantique tel que le Vérificateur accepte avec probabilité 1.*

Démonstration. Par hypothèse, le graphe admet un 3-coloriage. Nommons celui-ci C et $C(x)$ la couleur du noeud $x \in V$ selon ce coloriage. Le message du Prouveur sera composé des deux états $|\Psi\rangle$ et $|\Phi\rangle$ où $|\Psi\rangle = |\Phi\rangle = \frac{1}{\sqrt{n}} \sum_i |i\rangle|C(i)\rangle$. Étant donné ce message, analysons la probabilité de succès de chacun des tests.

1. **Test 1 :** La probabilité de succès du *swap-test* sur les états $|\Psi\rangle$ et $|\Phi\rangle$ est $\frac{1}{2} + \frac{|\langle\Psi|\Phi\rangle|^2}{2}$. Puisque les deux états sont identiques, cette valeur est 1 et la probabilité de succès de même.

2. **Test 2 :** Puisque C est un coloriage valide, il en découle que, pour tout résultat de la mesure effectuée sur les registres, les relations entre les noeuds seront satisfaites et donc que le Test 2 réussi avec probabilité 1.

3. **Test 3 :** Ce test s'assure que tous les noeuds sont présents dans la superposition. Pour mettre en évidence que sa probabilité de succès est 1, considérons cette équation :

$$(I \otimes F_3)\frac{1}{\sqrt{n}}\sum_j |j\rangle|c_j\rangle = \frac{1}{\sqrt{n}}\sum_j |j\rangle\frac{1}{\sqrt{3}}\sum_k e^{2\pi i c_j k/3}|k\rangle$$

On voit donc que si le résultat de la mesure sur la partie *couleur* correspond à $|0\rangle$ (et donc $k = 0$), la phase est la même pour chacun des j dans l'état résultant. Ceci correspond à $\frac{1}{\sqrt{n}}\sum_i |i\rangle = F_n|0\rangle$ et permet de conclure.

□

Un mot sur le Vérificateur est maintenant de rigueur. Il faut en effet s'assurer que ces trois tests puissent être effectués par un circuit quantique de taille polynomiale fonctionnant sur un nombre logarithmique de qubits et terminé par une mesure sur un seul qubit.

1. **Test 1 :** Le *swap-test* est défini comme le circuit de la figure 4.1.

   Il n'est donc pas nécessaire d'avoir plus d'un qubit auxiliaire et le circuit se fait en un nombre polynomial de portes.

2. **Test 2 :** Imaginons d'abord une mesure complète du système et décrivons un circuit classique sur cette entrée. Notons $N_{(a,b)}(x, y, p)$ un circuit qui, si $x = a$ et $y = b$ vérifie le coloriage de cette paire de noeuds ; si le coloriage n'est pas satisfaisant, le circuit a en sortie $(x, y, 0)$ sinon (x, y, p). Il suffit ensuite de mettre l'un à la suite de l'autre ces petits circuits. Le bit final du circuit entier est bien sûr p. Il sera nécessaire, afin de rendre ce circuit réversible, de reporter les bits d'entré à la sortie. Finalement, en remplaçant la mesure initiale par une pseudo-mesure nous obtenons un circuit unitaire. Celui-ci a une largeur logarithmique et une taille polynomiale.

3. **Test 3 :** Semblablement à ce qui a été décrit dans le Test 2, il est possible de remplacer chaque mesure dans la base de Fourier par une pseudo-mesure.

Suit un simple test logique sur la sortie de ces mesures. Tout ceci ne requiert pas plus qu'un nombre logarithmique de portes.

Penchons-nous maintenant sur la résilience du protocole qui constitue la plus grande partie de la preuve. Nous conclurons que, si $x \notin 3\mathcal{COL}$, alors la probabilité qu'a le Vérificateur de rejeter le mot est dans $\Omega(\frac{1}{n^6})$. Afin de conclure, nous aurons besoin des cinq lemmes suivants.

Mentionnons tout d'abord que la promesse contenue dans le protocole – les deux registres transmis par le Prouveur ne sont pas intriqués – nous permet d'écrire ceux-ci séparément sous la forme suivante :

$$|\Psi\rangle = \sum_i \alpha_i |i\rangle \sum_j \beta_{i,j} |j\rangle \qquad |\Phi\rangle = \sum_i \alpha'_i |i\rangle \sum_j \beta'_{i,j} |j\rangle$$

où $\sum_i |\alpha_i|^2 = 1$ et $\forall i \ \sum_j |\beta_{i,j}|^2 = 1$ et de même pour $|\Phi\rangle$. Nous exprimons ainsi les états afin de mettre en évidence l'amplitude correspondant à chaque noeud et celle, pour chacun d'eux, correspondant à chaque couleur.

Nous pouvons, sans perte de généralité, décrire chacun des registres comme un état pur. On peut en effet argumenter que, par linéarité, s'il existe un état mixte qui engendre une certaine probabilité de succès, l'un des états purs qui compose la mixture a au moins cette probabilité de succès. En particulier, puisque le protocole n'est constitué que d'un message du Prouveur, ce dernier ne peut prendre avantage d'intrication avec celui-ci.

Comme mentionné précédemment, l'utilisation de deux registres non intriqués nous permet de vérifier certaines caractéristiques de la preuve. Dans le présent protocole, le vérificateur s'attend à recevoir deux copies exactes du même état. Et tout au long de l'analyse de la résilience, nous capitaliserons sur cette similitude. Le premier test servira justement à vérifier que les deux registres sont semblables. Le lemme suivant énonce que si le Test 1 réussit, alors les deux registres sont semblables ; le Test 1fournit une mesure de cette similarité.

Lemme 5. *Soient* $|\Psi\rangle$ *et* $|\Phi\rangle$ *tels que précédemment définis. Si* $\exists k, l$ *tels que* $||\alpha_k\beta_{k,l}|^2 - |\alpha'_k\beta'_{k,l}|^2| \geq 1/n^3$ *alors le Test 1 échoue avec probabilité au moins* $\frac{1}{8n^6}$.

Démonstration. Soit $P_{i,j} = |\alpha_i\beta_{i,j}|^2$ et $Q_{i,j} = |\alpha'_i\beta'_{i,j}|^2$ les distributions de probabilités lorsque $|\Phi\rangle$ et $|\Psi\rangle$ sont mesurés dans la base de calcul. Nous faisons usage du fait que pour toute mesure de von Neumann $D(|\Psi\rangle, |\Phi\rangle) \geq D(P, Q)$ [NC00] pour les distances telles que définies plus bas ; P et Q sont les distributions classiques des résultats de mesures des états $|\Phi\rangle$ et $|\Psi\rangle$. Nous pouvons donc faire les calculs :

$$\begin{aligned}
\sqrt{1 - |\langle\Psi|\Phi\rangle|^2} &\stackrel{\text{def}}{=} D(|\Psi\rangle, |\Phi\rangle) \\
&\geq D(P, Q) \\
&\stackrel{\text{def}}{=} \frac{1}{2}\sum_{ij} \left||\alpha_i\beta_{i,j}|^2 - |\alpha'_i\beta'_{i,j}|^2\right| \\
&\geq \frac{1}{2}\left||\alpha_k\beta_{k,l}|^2 - |\alpha'_k\beta'_{k,l}|^2\right| \\
&\geq \frac{1}{2} \cdot \frac{1}{n^3}.
\end{aligned}$$

Nous pouvons donc conclure que $|\langle\Psi|\Phi\rangle|^2 \leq 1 - \frac{1}{4n^6}$ et qu'alors le Test 1 (lemme 4) échoue avec probabilité au moins $\frac{1}{8n^6}$. □

Tel que nous avons défini chacun des registres, nous voudrions que chaque noeud dans la superposition ait une couleur bien définie. Ceci se vérifie simplement avec la partie a) du second test. Si le même noeud est obtenu dans la mesure de chacun des deux registres, la même couleur doit être aussi mesurée dans chacun d'eux. Il est difficile de voir en quoi avoir une superposition de couleurs puisse aider un Prouveur malhonnête. Mais il nous est nécessaire, afin de poursuivre l'analyse du test, d'établir une borne inférieure sur l'amplitude d'une couleur pour chacun des noeuds dont l'amplitude est elle-même non négligeable.

Comme jusqu'à maintenant nous ne pouvons rien affirmer sur l'amplitude cor-

respondant à chacun des noeuds, le lemme suivant ne concerne donc que les noeuds dont celle-ci est non-négligeable.

Lemme 6. *Si la probabilité d'échec aux tests 1 et 2a) est inférieure ou égale à* $\frac{1}{8n^6}$*, alors* $\forall i$ *tels que* $|\alpha_i|^2 \geq \frac{1}{n^2}$ *il existe un et un seul* j *tel que* $|\beta_{i,j}|^2 \geq \frac{99}{100}$*.*

Démonstration. Par contradiction, supposons l'existence d'un i tel que $|\alpha_i|^2 \geq \frac{1}{n^2}$ pour lequel il existe deux j tels que $\beta_{i,j}$ a une norme supérieure à $1/200$. Ainsi, supposons, sans perte de généralité, que $|\beta_{i,0}|^2 > 1/200$ et $|\beta_{i,1}|^2 > 1/200$. Il découle du lemme 5 que $|\alpha_i'|^2 \left|\beta_{i,1}'\right|^2 \geq |\alpha_i|^2 |\beta_{i,1}|^2 - \frac{1}{n^3} \geq \frac{1}{200n^2} - \frac{1}{n^3}$. Il s'ensuit que la probabilité d'obtenir $(i,0)$ de la mesure du registre $|\Psi\rangle$ et $(i,1)$ de la mesure du registre $|\Phi\rangle$ est au moins

$$\left(\frac{1}{200n^2}\right)\left(\frac{1}{200n^2} - \frac{1}{n^3}\right) \geq \frac{1}{8n^6}.$$

pour n suffisamment grand. Ceci contredit l'hypothèse et nous permet de conclure. □

Étant donné qu'il ne peut y avoir qu'une couleur par noeud et que les deux registres doivent être semblables, que reste-t-il au Prouveur comme marge de manoeuvre pour tricher ? Il lui reste le plus évident, omettre les noeuds problématiques. Le Vérificateur voudra donc s'assurer que tous les noeuds font partie de la superposition et en fait que chaque noeud ait une amplitude semblable.

On se souviendra que le Test 3 ne peut se poursuivre que si un certain résultat de mesure est obtenu dans la partie *couleur* d'un registre. Le lemme qui suit s'attarde donc à spécifier avec quelle probabilité le Test 3 peut être concluant.

Lemme 7. *Étant donnée une probabilité d'échec aux tests 1, 2a) inférieure à* $\frac{1}{8n^6}$*, la probabilité d'obtenir une mesure correspondant à* $F_3|0\rangle := |\overline{0}\rangle$ *sur la partie* couleur *d'un registre est supérieure à* $1/5$ *pour* n *suffisament grand.*

Démonstration. Pour les besoins de la preuve, supposons que la partie *indice* soit déjà mesurée. Si le résultat de cette mesure est i, la probabilité d'obtenir $|\overline{0}\rangle$ sur la partie *couleur* du registre est

$$\frac{1}{3}|\beta_{i,0}+\beta_{i,1}+\beta_{i,2}|^2.$$

Pour tout i avec une probabilité plus grande que $1/n^2$ d'être mesuré, le lemme 6 nous permet de supposer, sans perte de généralité que $|\beta_{i,0}|^2 > 99/100$ et $|\beta_{i,1}|^2 + |\beta_{i,2}|^2 \leq 1/100$. Par l'inégalité de Cauchy-Schwarz, nous obtenons le résultat suivant :

$$\begin{aligned}\frac{1}{3}|\beta_{i,0}+\beta_{i,1}+\beta_{i,2}|^2 &\geq \frac{1}{3}|\ |\beta_{i,0}|-|\beta_{i,1}+\beta_{i,2}|\ |^2\\ &\geq \frac{1}{3}|\ |\beta_{i,0}|-\sqrt{2(|\beta_{i,1}|^2+|\beta_{i,2}|^2)}\ |^2\\ &\geq \frac{1}{4}.\end{aligned}$$

Notons qu'au plus $n-1$ noeuds peuvent avoir une probabilité inférieure à $\frac{1}{n^2}$ d'être obtenue lors de la mesure et donc d'échapper à l'analyse du lemme 6. La probabilité d'obtenir $|\overline{0}\rangle$ lors de la mesure de la partie *couleur* du registre est donc au moins $(1-(n-1)\frac{1}{n^2})\frac{1}{4} \geq \frac{1}{5}$ pour n suffisament grand. □

Le lemme suivant établit, étant donné que la première partie du Test 3 est positive, que si l'un des noeuds a une trop petite amplitude, le Test 3 échouera.

Lemme 8. *Soit $|X\rangle = \sum_i \gamma_i|i\rangle$, l'état résultant de la mesure de la partie* couleur *du registre. Si $\exists l$ tel que $|\gamma_l|^2 < \frac{1}{2n}$, alors la probabilité de ne pas obtenir $F_n|0\rangle$ lors de la mesure de $|X\rangle$ est au moins $\frac{1}{16n^2}$.*

Démonstration. Posons $|\overline{0}\rangle := F_n|0\rangle$. Soit P et Q les distributions de probabilités de la mesure dans la base de calcul de $|X\rangle$ et $F_n|0\rangle$ respectivement. Semblablement

à la technique utilisée dans le lemme 5 nous arrivons à :

$$\begin{aligned}
\sqrt{1-|\langle X|\overline{0}\rangle|^2} &\stackrel{\text{def}}{=} D(|X\rangle, |\overline{0}\rangle) \\
&\geq D(P,Q) \\
&\stackrel{\text{def}}{=} \frac{1}{2}\sum_i \left||\gamma_i|^2 - \frac{1}{n}\right| \\
&\geq \frac{1}{2}\left||\gamma_l|^2 - \frac{1}{n}\right| \\
&\geq \frac{1}{4n}.
\end{aligned}$$

Il s'ensuit que la probabilité de mesurer $|\overline{0}\rangle$ est au plus $1 - \frac{1}{16n^2}$ et donc celle d'échouer le test au moins $\frac{1}{16n^2}$. □

Finalement, ce lemme conclut à une borne inférieure pour la norme de l'amplitude de chaque noeud.

Lemme 9. *Si la probabilité de succès est supérieure à $\frac{1}{8n^6}$ pour les tests 1, 2a et 3, alors $\forall i$, $|\alpha_i|^2 \geq \frac{1}{10n}$.*

Démonstration. Le lemme 7 conclut que la probabilité de mesurer 0 dans la partie *couleur* d'un registre, lors du Test 3, est au moins 1/5. Soit $|X\rangle = \sum_i \gamma_i|i\rangle$ l'état résultant après avoir obtenu 0 lors de cette mesure.

Par contradiction, supposons qu'il existe un i pour lequel $|\alpha|^2 < 1/(10n)$. Suite au lemme 7 on peut conclure que $|\gamma_i|^2 < 1/(2n)$. Il s'ensuit, conjugué au lemme 8, que le Test 3 échoue avec une trop grande probabilité. Nous pouvons donc en conclure que, $\forall i$, $|\alpha|^2 \geq 1/(10n)$. □

Il est maintenant possible, en conjuguant les lemmes 5, 6, 7 et 9, de conclure sur la résilience du protocole. Tous les lemmes précédents ont établi certaines propriétés de la preuve fournie par le Prouveur. Mais jusqu'à maintenant, aucun n'a abordé la

colorabilité du graphe ! Le lemme suivant conclut donc que si la preuve est telle que la probabilité de succès de tous les tests jusqu'ici étudiés (1, 2a et 3) est grande, le Vérificateur réalisera, avec probabilité non négligeable, que le coloriage n'est pas valide le cas échéant.

Théorème 36. *Si $x \notin 3\mathcal{COL}$ le Vérificateur rejette le mot avec probabilité au moins $\frac{1}{24n^6}$.*

Démonstration. Soit $G \notin 3\mathcal{COL}$ et que la preuve est telle que la probabilité d'échec aux tests 1, 2a et 3 est plus petite que $\frac{1}{8n^6}$. Posons $C(i) = \max_j |\beta_{ij}|$ afin de définir un coloriage. Suite à l'analyse des lemmes 5 et 6, nous savons qu'un noeud a une amplitude plus grande que les autres et que, pour ce noeud, une couleur a elle aussi une plus grande amplitude. Ce maximum est donc bien défini. Puisque $G \notin 3\mathcal{COL}$, il existe deux sommets adjacents s_1 et s_2 tel que $C(s_1) = C(s_2)$. Le lemme 9 nous apprend qu'au Test 2, la probabilité de mesurer $C(s_1)$ dans le premier registre est au moins dans l'ordre de $1/(10n^2)$. Les lemmes 5 et 9 mènent à conclure que la probabilité de mesurer $C(s_2)$ dans le second registre est alors au moins dans l'ordre de $1/(10n^2) - 1/n^3$ Il s'ensuit donc que la probabilité qu'a le Vérificateur de vérifier cette arête est au moins $\frac{1}{8n^6}$ qui est donc aussi la probabilité d'échouer le Test 2b). □

4.3.2 Preuve classique de taille polynomiale

Afin de montrer l'équivalence des classes, il nous reste à montrer l'inclusion suivante :

Théorème 37. $\mathsf{PQMA}_{\log}(2) \subseteq \mathsf{NP}$

Démonstration. Pour ce faire, nous montrerons que pour tout langage de la classe $\mathsf{PQMA}_{\log}(2)$ il existe un Vérificateur classique fonctionnant en temps polynomial. Nous nommerons V' ce Vérificateur. Dans la définition même de $\mathsf{PQMA}_{\log}(2)$, on se

souviendra que le prétraitement et la construction du Vérificateur quantique V sont effectués par une machine de Turing en temps polynomial. Le Vérificateur V' est donc en mesure de faire exactement les mêmes opérations. Il reste à démontrer que V' est en mesure de simuler le circuit quantique qu'est V et ceci avec suffisamment de précision.

Dans la définition de la classe $\mathsf{PQMA}_{\log}(2)$, le Vérificateur V peut être décrit de façon classique. Pour un mot de taille n, nous considérerons un circuit de largeur $O(\log(n))$ où chacune des portes est décrite comme une matrice carrée de taille $O(n)$; l'identité est donc appliquée à l'espace des qubits qui ne sont pas impliqués dans la porte. Notons c le nombre maximal de dimensions impliqués dans une porte du circuit. De plus, nous considérerons que $c \geq 3$ pour fin d'analyse. Le Vérificateur V' simulera donc un nombre polynomial de portes en effectuant un nombre polynomial de multiplications de matrices de taille polynomial. Par définition, si $x \in L$, pour $L \in \mathsf{PQMA}_{\log}(2)$, il existe un message quantique m de taille $O(\log(n))$ tel que la probabilité d'accepter est 1 alors que si $x \notin L$, un tel état n'existe pas. Le Vérificateur V' considérera donc comme certificat un vecteur de taille $O(n)$ correspondant à la description de ce système.

Il s'agit maintenant de montrer que cette simulation peut être faite avec suffisamment de précision pour être conforme à la définition de la classe. Étant donné que la définition de la classe implique une différence de $d = \frac{1}{p(n)}$ entre la résilience et la complétude, on voudra que la simulation soit faite avec précision $\frac{d}{3}$.

Si l'on multiplie les nombres a et b qui ont une borne supérieure sur leur erreur relative de ϵ, le résultat sera $(a+a\epsilon)(b+b\epsilon) = ab+2ab\epsilon+ab\epsilon^2 \leq ab+3ab\epsilon$. L'erreur relative est donc $O(3\epsilon)$. Comme chaque porte n'agit que sur un nombre constant c de dimensions, chaque ligne de la matrice correspondante n'aura que c valeurs non nulles. Lors de la multiplication de la matrice avec le vecteur d'état, l'erreur se répétera au plus c fois et l'erreur résultante sera donc dans $O(3c\epsilon)$.

Cette analyse se répète pour chaque porte du circuit : $(a + a\epsilon)(b + 3bc\epsilon) =$

$ab + 3abc\epsilon + ab\epsilon + 3abc\epsilon^2 \leq ab + 9abc\epsilon$. Pour une ligne de la matrice, ceci sera repris c fois et l'erreur sera donc dans $O(3^2c^2\epsilon)$. Soit q le nombre de portes dans le circuit, un nombre polynomial en n. Après avoir effectué la multiplication de q matrices avec le vecteur d'entrée, l'erreur pour chaque valeur du vecteur sera donc dans $O(3^q c^q \epsilon) \subseteq O(c^q)$. Le Vérificateur utilisera donc, sur chaque entrée, un nombre de bits de précision q' de sorte que $O(c^q) \subset O(2^{q'})$; c'est à dire pour que l'erreur croisse plus lentement que la précision. La description du Vérificateur V spécifie qu'une mesure sur le premier qubit est effectuée à la fin du calcul et que le mot est accepté si le résultat est 1. Le Vérificateur V' voudra évaluer la probabilité de cette mesure qui correspond au projecteur Π_1. Cette probabilité est, pour le vecteur final $|\psi\rangle$, $\langle\psi|\Pi_1|\psi\rangle$. Étant donné les calculs précédents, cette valeur peut être évaluée avec une erreur inférieure à $\frac{d}{3}$. Le Vérificateur V' acceptera donc si et seulement si la probabilité d'acceptation estimée est supérieure à $1 - \frac{d}{2}$. Ceci conclut la preuve. □

CONCLUSION

Cette thèse présente l'un des résultats de recherche originaux du candidat : la caractérisation de la classe NP par une classe de preuves interactives quantiques. La présentation de ce résultat a été précédée d'une revue de littérature sur les preuves interactives classiques et quantiques. Le chapitre premier présente les classes de complexité usuelles permettant de situer les classes présentées par la suite. Le chapitre second présente les preuves interactives à un seul prouveur. Le troisième chapitre est, lui, consacré aux preuves interactives à plusieurs prouveurs. C'est finalement au chapitre quatrième que le paradigme des classes interactives comportant un *court* message a été présenté. La classe $\mathsf{PQMA}_{\log}(2)$ y a été définie et la preuve de son égalité avec NP présentée.

Ces résultats amènent bien entendu nombre de questions. Nous présentons maintenant quelques-unes d'entre elles.

Le résultat présenté dans cette thèse – la caractérisation de la classe NP – a été obtenu en considérant une résilience polynomialement proche de 1. Est-il possible de faire mieux ?

Question 11. *Quelle est la plus petite erreur atteignable pour la classe* $\mathsf{PQMA}_{\log}(2)$ *?*

En effet, il ne suffit pas de chercher à atteindre la plus petite résilience possible. Tout l'intérêt est d'atteindre la plus grande différence entre la complétude et la résilience. Il serait, par exemple, possible de considérer une complétude et une résilience constantes sans que la première soit 1 ni la seconde 0. Mais peut-on vraiment atteindre une constante pour cette différence ? Les avis tendent vers la négative et il a été mentionné comme argument [ABD$^+$08] que dans un tel cas, il se pourrait que $\mathsf{QMA}(2) = \mathsf{NEXP}$ alors que ceci semble improbable.

Nous avons mentionné, dans le chapitre précédent, qu'une résilience constante a été atteinte avec un message de taille $\sqrt{n}\mathsf{polylog}(n)$ [ABD$^+$08]. Ceci amène donc la question suivante :

Question 12. *Quelle est la plus petite taille de message permettant d'atteindre une complétude et une résilience constantes pour un langage* NP*-complet ?*

Nous n'avons pas, dans la question précédente, spécifié la classe interactive en question. C'est que la question est plus complexe qu'il n'y paraît. Jusqu'à maintenant, un seul type de message a été considéré : celui constitué d'un certain nombre de registres, sans intrication entre eux, tous de taille logarithmique. Déjà, il est intéressant de savoir quel est le plus petit nombre nécessaire de ces registres pour atteindre une complétude et une résilience constantes. On peut, toutefois, considérer une variante. Un certain nombre de registres mais de différentes tailles. Une voie prometteuse est, en particulier, de n'avoir que deux registres, un court et un long. Un test pourrait alors être appliqué entre le court et le long afin de vérifier que ce dernier est effectivement composé de plusieurs copies non intriquées du premier ; ceci plutôt que d'en faire la promesse dans la définition de la classe. Ces considérations compliquent grandement les comparaisons entre les différents résultats à venir. Et pour ajouter à cette complication, la plupart des résultats ne pourront être généralisés pour l'ensemble de la classe NP, mais seulement à certains langages. Ceci pour la même raison que pour la classe $\mathsf{QMA}_{\log}(\sqrt{\mathsf{n}}\mathsf{polylog}(\mathsf{n}))$: la réduction polynomiale fait en sorte qu'une fois appliquée, la taille du registre, pour cet exemple, dépasse celle spécifiée dans la classe.

Question 13. *Quelles sont la complétude et la résilience optimales pour la classe* $\mathsf{QMA}_{\log}(\mathsf{polylog}(\mathsf{n}))$ *?*

En effet, il est intéressant de considérer différents nombres de registres dans le cas où ceux-ci ne sont pas intriqués. En particulier, considérer polylog registres permettrait de généraliser le résultat à l'ensemble de la classe NP plutôt qu'à un langage spécifique.

Toutes ces questions ne considèrent qu'une classe définie avec un seul message. Qu'en est-il alors d'une classe comportant plusieurs messages ? Quelles seraient les conséquences de l'utilisation d'un message de taille logarithmique pour une classe telle que QIP(2) ou QIP ? On peut en fait se poser les questions précédemment énoncées pour nombre de classes. En particulier, il serait intéressant de se pencher sur la classe co-NP et comparer les résultats ainsi obtenus avec ceux qui concernent NP. Comme toute recherche un tant soit peu intéressante, celle-ci mène donc à plus de questions que de réponses.

BIBLIOGRAPHIE

[ABD+08] S. Aaronson, S. Beigi, A. Drucker, B. Fefferman et P. Shor. « The power of unentanglement ». Dans *Proceedings of IEEE Annual Conference on Computational Complexity*, pages 223–236, 2008.

[Bab85] L. Babai. « Trading group theory for randomness ». Dans *Proceedings of ACM symposium on Theory of computing*, numéro 17, pages 421–429, 1985.

[BBT05] G. Brassard, A. Broadbent et A. Tapp. « Quantum pseudo-telepathy ». *Foundations of Physics*, 35 :1877–1907, 2005.

[BCWW01] H. Buhrman, R. Cleve, J. Watrous et R. de Wolf. « Quantum fingerprinting ». *Physical Review Letters*, (87), 2001.

[BFL91] L. Babai, L. Fortnow et C. Lund. « Non-deterministic exponential time has two-prover interactive protocols ». pages 3–40, 1991.

[BGKW88] M. Ben-Or, S. Goldwasser, J. Kilian et A. Wigderson. « Multi-prover interactive proofs : how to remove intractability assumptions ». Dans *Proceedings of ACM symposium on Theory of computing*, numéro 20, pages 113–131, 1988.

[BHZ87] R. B. Boppana, J. Hastad et S. Zachos. « Does co-NP have short interactive proofs ». *Information Processing Letters*, 25 :127–132, 1987.

[BT07] H. Blier et A. Tapp. « All languages in NP have very short quantum proofs ». *arXiv :0709,0738v1 [quant-ph]*, 2007.

[BT08] H. Blier et A. Tapp. « A single initialization server for multi-party cryptography ». Dans *Proceedings of International Conference Information Theoretic Security*, pages 71–85, 2008.

[BT09a] H. Blier et A. Tapp. « All languages in NP have very short quantum proofs ». Dans *The Third International Conference on Quantum, Nano and Micro Technologies*, pages 34–37, 2009.

[BT09b] H. Blier et A. Tapp. « A quantum characterization of NP ». *Soumis à Computational Complexity*, 2009.

[CHTW04] R. Cleve, P. Hoyer, B. Toner et J. Watrous. « Consequences and limits of nonlocal strategies ». Dans *Proceedings of IEEE Annual Conference on Computational Complexity*, pages 236– 249, 2004.

[Deu85] D. Deutsch. « Quantum theory, the church-turing principle and the universal quantum computer ». Dans *Proceedings of the Royal Society*, numéro A400, pages 97–117, 1985.

[Fei91] U. Feige. « On the success probability of the two provers in one round proof systems ». *Structures*, pages 116–123, 1991.

[FL92] U. Feige et L. Lovász. « Two-prover one-round proof systems : their power and their problems (extended abstract) ». Dans *Proceedings of ACM symposium on Theory of computing*, pages 733 – 744, 1992.

[FRS88] L. Fortnow, J. Rompel et M. Sipser. « On the power of multi-power interactive protocols ». Dans *Proceedings of Structure in Complexity Theory Conference*, pages 156–161, 1988.

[GMR85] S. Goldwasser, S. Micali et C. Rackoff. « The knowledge complexity of interactive protocols ». Dans *Proceedings of ACM symposium on Theory of computing*, numéro 17, pages 291–304, 1985.

[Gol01] O. Goldreich. *Foundations of Cryptography.* Cambridge, 2001.

[GS86] Shafi Goldwasser et Michael Sipser. « Private coins versus public coins in interactive proof systems ». Dans *Proceedings of ACM symposium on Theory of computing*, numéro 18, pages 59–68, 1986.

[IKM08] T Ito, H Kobayashi et K Matsumoto. « Oracularization and two-prover one-round interactive proofs against nonlocal strategies ». *arXiv :0810.0693v1 [quant-ph]*, 2008.

[JJUW09] R. Jain, Z. Ji, S. Upadhyay et J. Watrous. « QIP = PSPACE ». quant-ph/0907.4737, 2009.

[Kil90] J. Kilian. « Strong separation models of multi-prover interactive proofs ». Dans *DIMACS Workshop on Cryptography*, 1990.

[KKM+08] J. Kempe, H. Kobayashi, K. Matsumoto, B. Toner et T. Vidick. « Oracularization and two-prover one-round interactive proofs against nonlocal strategies ». *arXiv :0810.0693v1 [quant-ph]*, 2008.

[KM03] H. Kobayashi et K. Matsumoto. « Quantum multi-prover interactive proof systems with limited prior entanglement ». *Journal of Computer and System Sciences*, pages 429–450, 2003.

[KW00] A. Kitaev et J. Watrous. « Parallelization, amplification, and exponential time simulation of quantum interactive proof systems ». Dans *Proceedings of ACM symposium on Theory of computing*, numéro 32, pages 608–617, 2000.

[LFKN92] C. Lund, L. Fortnow, H. Karloff et N. Nisan. « Algebraic methods for interactive proof systems ». *Journal of the Association for Computing Machinery*, 39(4) :859–868, 1992.

[LS91] D. Lapidot et A. Shamir. « Fully parallelized multi prover protocols for NEXP-time ». Dans *Proceeding of IEEE Symposium on Foundations of Computer Science*, pages 13–18, 1991.

[MW05] C. Marriott et J. Watrous. « Quantum arthur-merlin games ». *Computational Complexity*, (14) :122–152, 2005.

[NC00] M. Nielsen et I. Chuang. *Quantum Computation and Quantum Information*. Cambridge, 2000.

[Sha92] A. Shamir. « IP = PSPACE ». *Journal of the Association for Computing Machinery*, 39(4) :869–877, October 1992.

[Sto77] K. Stockmeyer. « The polynomial-time hierarchy ». *Theoretical Computer Science*, 3 :1–22, 1977.

[Wat99] J. Watrous. « PSPACE has constant-round quantum interactive proof systems ». Dans *Proceeding of IEEE Symposium on Foundations of Computer Science*, pages 112–119, 1999.

[Wat03] J. Watrous. « On the complexity of simulating space-bounded quantum computations ». *Computational Complexity*, (12) :48–84, 2003.

Annexe I

Diagramme d'inclusion

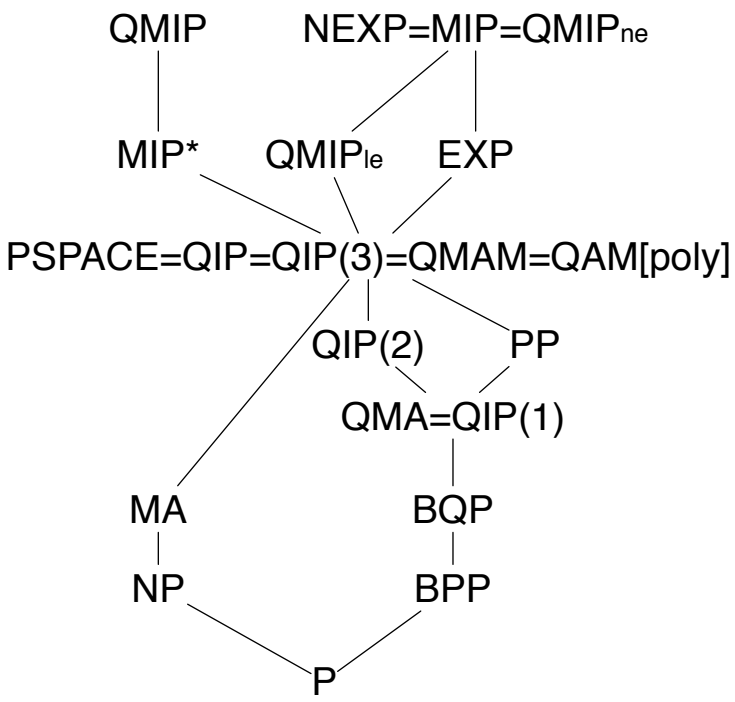

FIGURE I.1 – Diagramme d'inclusion