(Non)practicabilité de l'algorithme classique-quantique de factorisation des entiers

Razvan Barbulescu

(Non)practicabilité de l'algorithme classique-quantique de factorisation des entiers

Razvan Barbulescu

-rapport technique-

1 L'ordinateur quantique dans le crible de NFS : Shor pour l'étape de cofactorization

Le développement de l'informatique quantique peut aboutir à long terme à un ordinateur capable de casser RSA 1024. Mais cet objectif semble tellement éloigné que Preskill [Pre18] fait l'hypothèse suivante : pendant une longue période on disposera de dispositifs quantiques ayant entre 50 et 150 qubits, appelés en anglais noisy intermediate scale quantum (NISQ) object. Il faut donc concevoir et implanter des algorithmes ayant un nombre réduit de qubits. Voici une possible application. Le logiciel CADO-NFS [Tea17] utilise une partie non-négligeable du temps de calcul pour une étape appelé cofactorisation. Nous faisons l'exercice d'analyser l'accélération du temps total d'une factorisation avec CADO-NFS d'une clé RSA entre 60 et 180 chiffres décimaux sous l'hypothèse que la factorisation d'entiers entre 20 et 75 bits est plus rapide sur un ordinateur quantique que sur un dispositif classique.

1.1 Les faibles bases de notre hypothèse

L'algorithme de Shor [Sho99] est une descripion de haut niveau d'un algorithme qui admet plusieurs variantes de bas niveau, utisant 5n qubits dans la version la plus simple et 2n+2 dans la version optimisé pour le nombre de qubits [TK08]. Lors de la cofactorisation, CADO-NFS factorise des entiers de mfb0 et mfb1 bits, deux paramètres dont la valeur optimale est disponible dans le dossier "parameters/factor" du logiciel, Nous reproduisons ces valeurs dans le Tableau 1. On factorise le cofacteur plus petit et seulement dans une petite fraction de cas on factorise le cofacteur plus grand, donc un dispositif NISQ de $2 \cdot 62 + 2 = 126$ qubits permet d'implenter la cofactorisation par l'agorithme de Shor dans NFS jusqu'à 18 chiffres décimaux, sous notre hypothèse d'un NISQ de 150 qubits avec codes correcteurs.

Le rapport NIST sur l'informatique quantique [CJL+16] identifie plusieurs tâches de recherche qui, ensemble, permettrons la construction de l'ordinateur quantique :

dec. digits	60	70	80	90	100	120	140	160	180
mfb0 (bits)	17	19	41	46	49	54	58	60	62
mfb1 (bits)	35	42	42	46	50	54	58	84	99
$\log_2(N_g)$	58	65	71	83	89	90	101	118	131
$\log_2(N_f)$	104	110	120	122	129	150	183	208	215

Table 1: Taille binaire des entiers factorisés lors de la cofactorisation (NFS)

1. L'informatique quantique nécessite des codes correcteurs quantiques. Si le premier tel code, proposé par Shor [Sho96], était de type $(n, k, d) = (9, 1, 1)$, les codes récents [EKZ20] admettent k arbitrairement grand et $d \approx k/(\log k)$. L'objectif $d = ck$, pour une constante c, n'a pas été atteint mais les codes corecteurs quantiques sont suffisamment bons pour des petites calculs.

2. Le théorème du seuil [DA98], threshold theorem en anglais, donne le taux d'erreur des portes quantiques en dessous duquel on peut utiliser les codes correcteurs quantiques pour faire des longs calculs. Comme valeur numérique, [FSG09] estime le seuil à 1%.

3. Le taux d'erreur des portes quantiques varie selon la réalisation des qubits : ions figés (États-Unis : IonQ, Honeywell, etc), photons (France : Quandela, Chine), atoms neutres (France : Pasqual, etc), circuits supraconducteurs (États-Unis : IBM, Google, etc). Dans le cas des circuits supraconducteurs, l'application d'une portes quantiques consiste à envoyer un signal électromécanique à un circuit électronique non-linéaire. De nombreux travaux de contrôle optimal ont augmenté la fidélité des portes : plus de 99.9% pour les portes à 1 quibit, environ 1% pour les portes à 2 quibts [KWS$^+$20]. Nous sommes donc tout juste au au-delà du seuil, mais le domaine est très actif.

4. Les ordnateurs quantiques doivent regrouper les résultats des tâches précédentes. Or, les plus grands ordinateurs actuels soit ne sont pas programmables [ZWD$^+$20, Whi] soit n'utilisent pas de codes correcteurs [Mar21, JJAB$^+$21, MPD$^+$20]. Les travaux d'implantation de codes correcteurs sont faits sur des ordinateurs plus petits, le domaine étant actif [CSA$^+$21].

Un bref bilan est que les tâches (1)-(4) n'ont pas été accomplies, mais elles peuvent l'être dans le futur proche. Cela conditionne une estimation précise des resources quantiques nécessaires pour implanter l'algorithme de Shor pour un entier de taille moyenne, entre 20 et 75 bits. Nous concluons qu'il est trop tôt pour décider si notre hypothèse est réaliste. La motivation relève ainsi premièrement de la volonté de comprendre le comportement du logiciel CADO-NFS dans le cas où la communauté des chercheurs fait du progrès sur

dec. digits	60	70	80	90	100	110	120
CPU lin. alg.(s)	2.2	4.0	16.2	99.04	556	7737	6289
CPU crible (s)	17.3	31.5	176.4	780.0	2715	1351	25562
crible/total (%)	88.5	88.7	91.6	88.7	83.0	85.1	80.3

Table 2:

l'étape de cofactorisation, indépendamment de la raison (matériel, algorithme, implémentatin, etc).

1.2 Expériences avec CADO-NFS

Nous rappelons brièvement les grandes lignes de l'algorithme NFS (voir [LLMP93] pour une référence plus ample) :

1. polyselect : choisir deux polynômes bivariés homogènes $F, G \in \mathbb{Z}[x, y]$ satisfaisant une série de contraintes, par défaut CADO-NFS choisit une pair (F, G) avec G linéaire;
2. crible : énumérer un grand nombre de paires $(a, b) \in \mathbb{Z}^2$ telles que $N_g := G(a, b)$ est 2^{lpb0} et $N_f := F(a, b)$ est 2^{lpb1}; Les paires (a,b) sont groupées avant le calcul en paquets appelés special-q. Ceci est fait en deux étapes :
   (a) crible proprement dit : un procédé similaire au crible d'Erathostène produit la liste des paires (a, b) telles que $G(a, b)$ (resp. $F(a, b)$) divisé par les facteurs premiers inférieurs à 2^{lim0} (resp. 2^{lim1}), appelé cofacteur, est inférieur à 2^{mfb0} (resp. 2^{mfb1}).
   (b) cofactorisation : appliquer ECM pour factoriser les cofacteurs et collecter les cofacteurs ayant tous les facteurs inférieurs à 2^{lpb0} (resp. 2^{lpb1});
3. algèbre linéaire : écrire une matrice creuse ayant une ligne pour chacune des $2^{\text{lpb0}} + 2^{\text{lpb1}}$ paires (a, b) trouvées et calculer un vecteur non-nul du noyau;
4. racine carrée : finaliser la factorisation (temps négligeable).

Nous avons effectué des factorisations d'entiers entre 60 et 120 de chiffres décimaux. Pour tous les jeux de paramètres essayés, le crible représente plus de 80% du temps total, comme résumé dans le Tableau 1.2.

Selon les paramètres optimaux choisis par défaut par CADO-NFS, le temps du crible (étape 2) est dominé par le crible proprement dit (étape 2.a). Nous étudions comment évolue le rapport entre le temps des étapes 2.a et 2.b quand on modifie les paramètres mfb0 et mfb1. En augmentant les paramètres mfb on soumet plus d'entiers à la cofactorisation et on obtient plus de relations ppar special-q. Finalement, une meilleure efficacité par special-q permet de réduire le nombre de special-q, donc le coût total du crible proprement-dit (2.a)

(lpb0,lpb1,mfb0,mfb1)	CPU/sq		rels/sq	#sq	total CPU (s)
	2.a	2.b			2.a
(21,22,41,42,11)	1.1	0.6	190	1135	58.9
(21,22,55,55,11)	0.1	0.1	200	1095	70.9
(21,22,∞,∞,11)	—	—	≈ 300	—	—

Table 3: Réglage des paramètres de CADO-NFS pour un module de 70 chiffres décimaux. Estmation en Magma du nombre de paires (a,b) sans contraintes mfb.

diminue. En contre-partie, le coût de la cofactorisation (étape 2.a) augmente mais c'est le but de notre exercice d'investiguer la situation où la cofactorisation est négligeable, ou du moins acélérée par rapport à l'état actuel en CADO-NFS.

L'exemple d'un entier de 70 chiffres décimaux Dans le Tableau 3 nous comparons les résultsts de CADO-NFS si on augmente la proportion d'entiers qui sont envoyés à la cofactorisation. Cela est fait en augmentant les paramètres mfb0 et mfb1 de CADO-NFS. On remarque que pour mfb0=mfb1=55 le crible effectue moins de special-q donc moins de crible proprement-dit. Les approximations du calcul et le fait que l'implantation n'est pas optimisée pour cette situation limite où la cofactorisation est accélérée font que cela ne se traduit pas par une diminution du temps total de l'étape 2.a. À cela s'ajoute le fait que le logiciel n'accepte pas mfb ≥ 60 donc on ne peut pas faire une expérimentation qui trouve toutes les paires (a,b) existentes dans le domaine de crible.

La dernière ligne du tableau fait une estimation des résultats qu'on doit obtenir en choisissant une valeur mfb supérieure à la valeur de la taille binaire estimée des deux normes. Nous avons fait des expériences en Magma sur un échantillon d'un million de paires (a,b) du domaine de crible et constaté qu'approximativement 1/3 des paires ne sont pas trouvée si on choisit mfb comme dans sa valeur optimale proposée par CADO-NFS.

On conclue que le gain lié à augmenter mfb et à utiliser Shor lors de la cofactorisation gagnera dans le cas idéal quelques dixaines de pourcents. Une équipe innovante pourrait faire que le record de vitesse pour la factorisation d'entiers de 70 chiffres décimaux implique un circuit quantique.

1.3 Obstacles en pratique pour l'algorithme de Bernstein, Biasse et Mosca

Notre étude correpond à un cas particulier de l'algorithme de Bernstein, Biasse et Mosca [BBM17]. En effet, l'algorithme consiste à voir l'algorithme de Short comme un circuit quantique qui retourne 1 si F(a,b) et G(a,b) sont friables et 0 sinon. Ainsi, au lieu de cribler sur un domaine de taille S en temps proportionnal à S, on collecte les paires (a,b) friables en faisant $O(\sqrt{S})$ appels à Shor. Le cas limite quand N=1 correspond à ne pas appliquer Grover mais seulement à utiliser Shor dans la cofactorisation. Malgré sa meilleur analyse asymptotique,

l'algorithme risque de rester moins rapide que NFS en pratique. Nous relevons deux arguments contre cet algorithme.

Surcoût des codes correcteurs Ce point est évoqué par les auteurs même dans [BBM17]. Le nombre de qubits physiques requierts pour implanter n qubits logiques sur une profondeur T est égal à $O(n \cdot \text{polylog}(nT))$ (cf. [Got14]). Dans cet algorithme $\log T = (\log N)^{2/3+o(1)}$ donc le nombre de qubits physiques est $(\log N)^c$ pour une constante c issue du thèorème du seuil. Si $c \geq 1$, le nombre de qubits est $\log N$ et alors on peut appliquer directement Shor sur le module N à factoriser. En pratique aussi le nombre de qubits est grand. En effet, on ne factorise pas les cofacteurs mais les normes complètes donc on utilise au moins $2\log_2 N_f + 2\log_2 N_g + 4$ bits pour appliquer Shor sur chacune des normes en parallèle. Par exemple, pour factoriser un module de 180 chiffres décimaux on a $\log_2 N_f = 215$ et $\log_2 N_g = 131$. Cela est plus grand que la taille des cofacteurs mfb0=62 et mfb1=99 (voir Tableau 1).

Grover est incompatible avec le crible proprement-dit L'algorithme de Grover ne prend pas en entrée une liste mais seulement un ensemble de la forme $\{0,1\}^n$. Cela est suffisant pour la description théorique de l'algorithme de Bernstein et al. Mais Grover exclue la variante suivante : on fait du crible proprement-dit et on garde seulement un petit nombre de paires (a,b) pour la cofactorisation. Le gain de l'algorithme de Grover est au plus $\sqrt{S}$ où S est le nombre de paires d'un secial-q donc $\sqrt{D} = 2^I$ où $10 \leq I \leq 16$ est un paramètre de CADO-NFS. Ce gain est contrebalancé par la perte due au fait qu'on factorise plus d'entiers : la proportion de paires (a,b) qui sont soumises à la cofactorisation est entre 1/100 et 1/1000, ce qui élimine l'avantage de l'algorithme de Grover. Pour une analyse numérique complète il faut prendre en compte aussi le coût du crible proprement-dit, mais le point important est qu'il reste comparable à celui de la cofactorisation, comme montré dans le Tableau 3.

1.4 Conclusion

Les travaux de la communauté scientifique ont abouti sur des ordinateurs quantiques de plus de 50 qubits. Ceux-ci ne permettent pas d'implanter l'algorithme de Shor car ils n'implante pas de codes correcteurs d'erreurs mais cela est envisageable à court terme. Seulement un tel résultat permettra d'estimer le temps requiert par l'algorthme de Shor pour factoriser un entier de n bits, avec $n \approx 1/2$ du nombre de qubits.

Dans le cas où l'on estime un avantage quantique pour des entiers ayanat entre 17 et 62 bits, cela permettra dans le cas idéal seulement quelque dixaines de pourcent d'avantage quantique pour des modules RSA entre 60 et 100 chiffres décimaux. Cela n'est pas suffisant pour avoir des conséquences en cryptographie.

Le seul résultat qui combine NFS et Shor que nous avons pu trouver dans la litérature, l'algorithme de Bernstein et al., n'est pas faisable en pratique. Son principal problème est la profondeur du circuit quantique, ce qui nécessite

des codes correcteurs très performants et par conséquent une augmentation considérable du nombre de qubits physiques par rapport aux qubits logiques.

References

[BBM17] Daniel J Bernstein, Jean-François Biasse, and Michele Mosca. A low-resource quantum factoring algorithm. In *International Workshop on Post-Quantum Cryptography – PQCrypto*, volume 10346 of *LNCS*, pages 330–346. Springer, 2017.

[CJL+16] Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. Report on post-quantum cryptography (NISTIR 8105), 2016. Available online at `https://csrc.nist.gov/publications/detail/nistir/8105/final`.

[CSA+21] Zijun Chen, Kevin J Satzinger, Juan Atalaya, Alexander N Korotkov, Andrew Dunsworth, Daniel Sank, Chris Quintana, Matt McEwen, Rami Barends, Paul V Klimov, et al. Exponential suppression of bit or phase flip errors with repetitive error correction. *arXiv preprint arXiv:2102.06132*, 2021.

[DA98] Michael Ben-Or Dorit Aharonov. Fault-tolerant quantum computation with constant error rate, 1998. Avalable online at `https://arxiv.org/abs/quant-ph/9906129`.

[EKZ20] Shai Evra, Tali Kaufman, and Gilles Zémor. Decodable quantum LDPC codes beyond the square root distance barrier using high dimensional expanders. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 218–227. IEEE, 2020.

[FSG09] Austin G. Fowler, Ashley M. Stephens, and Peter Groszkowski. High-threshold universal quantum computation on the surface code. *Physical Review A*, 80(5), Nov 2009.

[Got14] Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *Quantum Information & Computation*, 14(15-16):1338–1372, 2014.

[JJAB+21] Petar Jurcevic, Ali Javadi-Abhari, Lev S Bishop, Isaac Lauer, Daniela F Bogorin, Markus Brink, Lauren Capelluto, Oktay Günlük, Toshinari Itoko, Naoki Kanazawa, et al. Demonstration of quantum volume 64 on a superconducting quantum computing system. *Quantum Science and Technology*, 6(2):025020, 2021.

[KWS+20] A Kandala, KX Wei, S Srinivasan, E Magesan, S Carnevale, GA Keefe, D Klaus, O Dial, and DC McKay. Demonstration

of a high-fidelity CNOT for fixed-frequency transmons with engineered ZZ suppression. Available online at `https://arxiv.org/abs/2011.07050`, 2020.

[LLMP93] Arjen K Lenstra, Hendrik W Lenstra, Mark S Manasse, and John M Pollard. The number field sieve. In *The development of the number field sieve*, pages 11–42. Springer, 1993.

[Mar21] John Martinis. Quantum supremacy using a programmable superconducting processor. *Bulletin of the American Physical Society*, 2021.

[MPD+20] Steven Moses, Juan Pino, Joan Dreiling, Caroline Figgatt, John Gaebler, Michael Allman, Charles Baldwin, Michael Foss-Feig, David Hayes, Karl Mayer, et al. Demonstration of the qccd trapped-ion quantum computer architecture. *Bulletin of the American Physical Society*, 65, 2020.

[Pre18] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, 2018.

[Sho96] Peter W Shor. Fault-tolerant quantum computation. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 56–65. IEEE, 1996.

[Sho99] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[Tea17] The CADO-NFS Development Team. CADO-NFS, an implementation of the number field sieve algorithm, 2017. Release 2.3.0.

[TK08] Yasuhiro Takahashi and Noboru Kunihiro. A fast quantum circuit for addition with few qubits. *Quantum Information & Computation*, 8(6):636–649, 2008.

[Whi] Jed Whittaker. System roadmap (report on the results of the d-wave team.

[ZWD+20] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020.