Jak używać algorytmu Grovera

Przewodnik po nauczaniu informatyki kwantowej cz. 7.

Marek Perkowski

absolwent Wydziału Elektroniki Politechniki Warszawskiej, tu również zdobył tytuł doktora automatyki. Od 1983 r. pracuje na Wydziale Inżynierii Elektrycznej i Komputerowej w Portland State University, gdzie jest profesorem zwyczajnym i dyrektorem Laboratorium Robotów Inteligentnych. Jeden ze współautorów WARP – pierwszego kompilatora języka VHDL dla układów FPGA. Twórca Diagramów Decyzyjnych Kroneckera, struktury krat logicznych i koncepcji robotów kwantowych. Przyczynił się do powstania oprogramowania dla syntezy logicznej, używanego w przemyśle USA. Pracował jako profesor wizytujący w Holandii, Francji, Japonii, Korei Południowej i Ludowej Republice Chin. W latach 2002–2004 był profesorem zwyczajnym w KAIST – Korean Advanced Institute of Science and Technology, gdzie zajmował się robotyką humanoidalną i komputerami kwantowymi. Kierował Komitetem Logiki Wielowartościowej IEEE w latach 2003–2005 i grupą roboczą Towarzystwa Inteligencji Obliczeniowej IEEE dla Inżynierii Kwantowej w latach 2006–2007. Autor ponad 515 publikacji o automatycznym projektowaniu, syntezie logicznej, logice wielowartościowej, logice odwracalnej, uczeniu maszynowym, robotyce i informatyce kwantowej.

Źródło: GetReal-WordPress.com

W tym numerze publikujemy ostatnią część „Przewodnika po nauczaniu informatyki kwantowej". Poprzednie można znaleźć w numerach 2-4/2021 Biuletynu PTI i we wszystkich tegorocznych wydaniach „Domeny". Wkrótce na stronie pisma opublikujemy cały „Przewodnik" w nadziei, że wówczas korzystanie z tego kompendium wiedzy będzie wygodniejsze.

Proces syntezy wyroczni zawiera etapy porównywalne do klasycznego projektowania układów cyfrowych przy użyciu narzędzi automatycznego projektowania: projekt architektury, synteza wysokiego poziomu, optymalizacja standardowych bloków (takich jak bloki arytmetyczne czy komparatory), synteza logiczna i layout kwantowy, czyli próba optymalnego rozmieszczenia bloków i bramek kwantowych w przestrzeni fizycznej realnego komputera kwantowego.

Jednym z wariantów jest LNNM (Linear Nearest Neighbor Model)[1], czyli liniowe rozmieszczenie kubitów, gdzie każdy kubit komunikuje się tylko z sąsiadami z góry i z dołu. Inny obecnie istotny wariant to rozmieszczenie kubitów w przestrzeni dwuwymiarowej, gdzie na przykład każdy kubit komunikuje się z sąsiadami z góry, z dołu, z lewej i prawej strony. Problem, jak optymalnie mapować układy kwantowe z bramek teoretycznych na kubity rozmieszczone regularnie w dwuwymiarowej przestrzeni opisanej grafem, w którym węzły są kubitami a krawędzie – parami kubitów, wymieniających informacje bezpośrednio między sobą. Przykład syntezy sumatora kwantowego dla layoutu liniowego można znaleźć w pracy[2].

Problem syntezy logicznej polega na tym, jak optymalnie zrealizować poszczególne bloki jako funkcje boolowskie. Ponieważ w naszym podejściu bloki te na ogół nie

1 Perkowski, M., Lukac, M., Shah, D., Kameyama, M.: *Synthesis of quantum circuits in Linear Nearest Neighbor model using Positive Davio Lattices*. Facta Universitatis, Ser.: Electronics and Energetics, 2011, 24(1), 73–89.

2 Dereizadeh, S., Premaratne, S.P., Matsuura, A.Y., Perkowski, M.: *Designing Gates to Realize a Full Adder Quantum Circuit in cQED Transmon Systems,* Book Chapter 20 in *Handbook: Nanoengineering, Quantum Sciences and Nanotechnologies*, Editor: Sergey Lyshevski, CRC Press by Taylor and Francis Group, LLC, 2019.

są funkcjami odwracalnymi (permutacyjnymi mapami jeden-na-jeden), wymagają one dodatkowych kubitów.

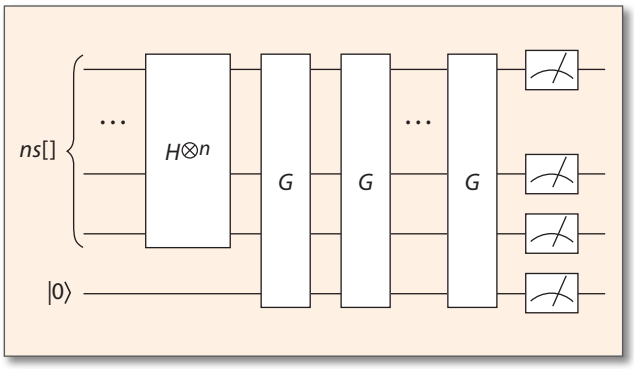
> *Nasza metodologia redukuje koszt kwantowy układu kosztem zwiększonej liczby kubitów. Praktycznie jest to jedyne możliwe rozwiązanie, gdyż projektant nie zna explicite funkcji boolowskiej, którą realizuje – zna jedynie bloki funkcjonalne, które się na nią składają.*

Jest to podobne do projektowania klasycznego komputera, który nigdy nie jest opisany w całości jako jeden automat. Omawiamy tylko niektóre zadania potrzebne do budowy wyroczni, ale nasze przykłady wystarczająco ilustrują podstawowe problemy, które musi rozwiązać projektant praktycznie realizowalnej wyroczni w językach opisu sprzętu kwantowego.

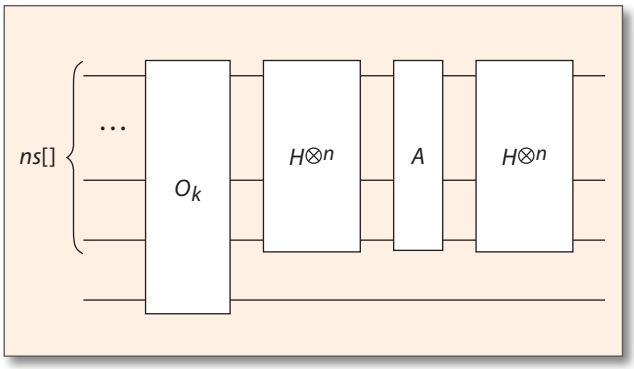
Bloki standardowego algorytmu Grovera

Ideą algorytmu Grovera jest umieszczenie kubitów reprezentujących całą przestrzeń poszukiwania o rozmiarze N w stanie superpozycji, praktyczna równoległość dla n zmiennych problemu, gdzie $N \leq 2^n$. Wyrocznia razem z następującym po niej operatorem dyfuzyjnym nazywana jest pętlą Grovera. Za transformatą Hadamarda (w kolejności przetwarzania) umieszczona jest wyrocznia. Jako pierwszy element pętli Grovera powoduje ona zmianę znaku stanów wyróżnionych (rozwiązań – mintermów z ON) na ujemny. Następnym elementem pętli Grovera jest operator dyfuzyjny, który zwiększa amplitudę stanów wyróżnionych, zwiększając tym samym prawdopodobieństwo, że te stany będą rezultatem pomiaru na wektorze kubitów, odpowiadającym zmiennym wejściowym. Po $O\sqrt{N}$ iteracjach pętli Grovera prawdopodobieństwo pomiaru stanu wyróżnionego jest zbliżone do 1 (w przypadku problemu posiadającego jedno rozwiązanie).

Wyrocznia klasyczna zmienia dla stanów wyróżnionych wartość boolowską bitu decyzyjnego. Natomiast wyrocznia kwantowa działa na całym stanie kwantowym wszystkich kubitów wejściowych i zmienia fazę kwantową na ujemną dla tych mintermów, dla których funkcja boolowska wyroczni jest spełniona.



Rys.1. Algorytm Grovera startuje z transformacji Hadamarda, po której następuje liczba $\frac{\pi}{4}\sqrt{\frac{N}{M}}$ wywołań G, po czym następuje pomiar kwantowy. Przez ns oznaczamy wektor zmiennych wejściowych problemu



Rys. 2. Schemat pętli Grovera G, składającej się z wyroczni dla danego problemu O_k, po której następuje operator dyfuzyjny.

Operując na kubitach, poszukiwanie Grovera używa wyroczni O_k i dokonuje szeregu iteracji pętli Grovera (opisanych jako G na rys. 1), aby znaleźć jedną z kombinacji wartości funkcji będących rozwiązaniami. Rys. 1 pokazuje bloki użyte w algorytmie Grovera, inicjalizację transformatą Hadamarda, następnie wielokrotne wywołanie pętli Grovera i na końcu pomiar. Bloki kwantowe na niższym poziomie hierarchii opisu, wewnątrz pętli Grovera, pokazuje rys. 2. Widać tu operator dyfuzji, który składa się z wektora bramek Hadamarda, bloku A i następnego bloku bramek Hadamarda. Blok A zawiera układ zmiany fazy zbudowany z wielowejściowej bramki Toffoliego, której kubit wyjściowy otoczony jest bramką Hadamarda po obu stronach. Układ dyfuzji może być zatem zbudowany jedynie z bramek odwracalnych i bramek Hadamarda.

Nie wyjaśniamy wszystkich detali algorytmu Grovera. Ważne jest jednak zauważenie, że kiedy żadna z kombina-

3 Rieffel, E., Polak, E.: *Quantum Computing: A Gentle Introduction*. The MIT Press, 2011.

4 Brassard, G., HØyer, P., Tapp, A.: Quantum counting. Automata, Languages and Programming, 1998, 820–831, DOI: 10.1007/BFb0055105

cji wejściowych nie spełnia warunków (funkcja boolowska wyroczni równa 0, niespełnialna), to algorytm zwróci losowo arbitralną kombinację wejściową. Zatem wszystkie odpowiedzi algorytmu powinny być weryfikowane na komputerze klasycznym albo na komputerze kwantowym poprzez sprawdzenie ich na wyroczni traktowanej teraz jak funkcja boolowska, użyta poza algorytmem Grovera. Jest to powtarzane wielokrotnie w algorytmach hybrydowych. Metafora algorytmu kwantowego jako „inteligentnego zgadywacza, który czasem się myli" jest poprawna i użyteczna.

Znajomość algorytmów

Dla większości praktycznych problemów istnieje wiele rozwiązań. Oznaczmy ich liczbę przez M. W takich przypadkach algorytm Grovera wymaga $\frac{\pi}{4}\sqrt{\frac{N}{M}}$ wywołań pętli Grovera, gdzie *N* jest rozmiarem przestrzeni poszukiwania (czyli liczbą elementów, na których dokonywane jest poszukiwanie), a M jest ilością elementów, które dają 1 w wyniku pomiaru (rozwiązania). Jest to złożoność czasowa, która dla małych M jest proporcjonalna do pierwiastka z N, dlatego mówimy o kwadratowym przyspieszeniu. Dowód można znaleźć w podręczniku Rieffela i Polaka[3]. Formuła ta wyjaśnia, dlaczego ważne jest, aby znać dokładnie albo przynajmniej umieć oszacować liczbę rozwiązań M. Algorytm liczenia kwantowego[4] może być użyty do dokładnego wyliczenia wartości M. Ten kwantowy algorytm wykorzystuje algorytm Grovera i kwantowy „algorytm estymacji fazy". Oba te algorytmy mają wiele zastosowań, zalecane jest zatem, żeby programista kwantowy zapoznał się z obydwoma tymi algorytmami i zrozumiał, jaka jest zasada ich stosowania do rozwiązywania różnych praktycznych problemów. Są to dwa najbardziej przydatne algorytmy kwantowe.

W problemach optymalizacyjnych istnieje M obiektów, z których każdy spełnia wszystkie ograniczenia i daje wartość wyjścia 1. Chcemy znaleźć taki obiekt, któremu odpowiada optymalna (maksymalna lub minimalna) wartość funkcji kosztu CF. Załóżmy, że chcemy minimalizować funkcję kosztu, co pozwoli nam odwołać się znowu do problemów minimalizacji boolowskiej. Problemy takie mogą być rozwiązywane poprzez wielokrotne wywoływanie problemów decyzyjnych w sekwencji wywołań algorytmu Grovera ze zmodyfikowanymi wyroczniami, które uwzględniają koszt poprzednio znalezionych rozwiązań nieminimalnych. Istnieje wiele znanych metod dokonania tej konwersji problemów optymalizacyjnych. Jedną z nich jest klasyczne poszukiwanie wykładnicze, które w istocie dokonuje zmodyfikowanej wersji poszukiwania binarnego na listach. Zauważmy, że poszukiwanie wykładnicze wymaga liczby porównań proporcjonalnej do logarytmu pozycji poszukiwanego obiektu. W kontekście problemów optymalizacyjnych „pozycja" obiektu jest jego wartością. Wykładnicze poszukiwanie identyfikuje obiekt z najmniejszą wartością funkcji celu spośród tych, które rozwiązują problem PSO, i obiekt ten jest rozwiązaniem problemu optymalizacyjnego.

Proces ten jest czasochłonny, jednak używanie algorytmu Grovera do rozwiązania problemów decyzyjnych w czasie $O(\sqrt{N})$ poważnie redukuje złożoność czasową, pod warunkiem że wyrocznia jest tak zaprojektowana, by weryfikowała rozwiązanie w czasie wielomianowym. Więcej o poszukiwaniu kwantowym można znaleźć w pracy[5].

Zakończenie

Problemy optymalizacyjne mogą być zredukowane do wielokrotnego powtarzania problemów decyzyjnych, takich jak PSO z wyroczniami modyfikowanymi przy każdej iteracji problemu PSO. Iteracje te wywołują algorytm Grovera z wyroczniami modyfikowanymi lub zmienianymi przez sterujący klasyczny procesor w hybrydowym systemie. Każdy problem PSO i odpowiedni problem optymalizacyjny może być rozwiązany z użyciem podanej metodologii. Ma to jednak sens jedynie wtedy, gdy nie jest znana żadna inteligentniejsza metoda poszukiwania rozwiązania od ślepego przeglądu pełnej przestrzeni w odpowiednim algorytmie klasycznym. Zwróćmy zatem uwagę, że algorytm Grovera jest przydatny dla tej „najgorszej" klasy problemów, gdy nie istnieje żadna metoda czy heurystyka. Algorytm Grovera jest też świetnym wstępem do tworzenia nowych algorytmów kwantowych.

Nie wiadomo, jak algorytmy oparte na wyroczniach, a także kwantowe sieci neuronalne będą sobie radziły w erze komputerów klasy NISQ (Noisy Intermediate Scale Quantum)[6], czyli w wolnym tłumaczeniu Zaszumionych Komputerów Kwantowych Pośredniego Poziomu. Mają być one w niedługim czasie dostępne. Jest to przejściowy model technologii komputerów bez wbudowanych systemów korekcji błędów. Będą jednak umiały rozwiązywać problemy redukowalne do układów, w których indywidualna precyzja na przykład operatorów arytmetycznych niekoniecznie będzie prowadzić do dużych błędów w wynikach algorytmów uczenia czy optymalizacji.

5 M. Boyer, M.: *Tight Bounds on Quantum Searching*. Proc. Workshop Physics and Computation (PhysComp96), New England Complex Systems Inst., Cambridge, Mass., 1996, 36–43.

6 Preskill, J.: *Quantum Computing in the NISQ era and beyond*. arXiv: 1801.00862v3 [quant-ph] 31 Jul 2018.