Wiesław CITKO, Wiesław SIEŃKO

Uniwersytet Morski w Gdyni, Katedra Telekomunikacji Morskiej

doi:10.15199/48.2019.09.30

Realizowalność algorytmów kwantowych z zastosowaniem opartych na sieciach neuronowych modeli uczenia maszynowego

Streszczenie. *W pracy zaproponowano zastosowanie modelu uczenia maszynowego do realizacji zadania faktoryzacji liczb całkowitych na iloczyn liczb pierwszych. Podano dwa algorytmy faktoryzacji oraz zaprezentowano model procesora analogowego realizującego powyższe zadanie. Efektywne metody faktoryzacji mają istotne znaczenie w łamaniu szyfrów opartych na systemie kryptograficznym RSA.*

Abstract*. In this paper a model for factorization of integer numbers on the product of prime numbers is presented. Two algorithms of factorization and a model of analog processor accomplishing the task of factorization are provided. Effective factoring methods are important in breaking codes on the RSA cryptographic system.* (***The Feasibility of Quantum Algorithms Using Machine Learning Models)***

Słowa kluczowe: uczenie maszynowe, kryptografia, algorytmy kwantowe, faktoryzacja liczb.
Keywords: machine learning, cryptography, quantum algorithms, factorization algorithms.

Wstęp

Algorytmy kwantowe takie jak algorytm Deutsch'a, Deutsch'a–Jozsa, Simon'a, Grover'a i w szczególności algorytm Shor'a stały się przesłanką do badań nad fizycznymi strukturami komputerów kwantowych. Podstawową cechą komputerów kwantowych, będących obiektami kwantowymi, jest istnienie stanu określonego jako paralelizm kwantowy (quantum parallelism). Dysponując zatem komputerem kwantowym tzn. fizycznym kwantowym obiektem można by znacznie przyśpieszyć wykonywanie w/w algorytmów w porównaniu z implementacjami klasycznymi [1]. W 1994 roku Peter Shor pokazał jak dysponując komputerem kwantowym można dokonać faktoryzacji liczby całkowitej za pomocą algorytmu o złożoności P-T (polynomial-time) [2]. Faktoryzacja liczb całkowitych jest problemem o znaczeniu praktycznym ze względu na jej zastosowanie w systemie kryptograficznym RSA. Jak wiadomo algorytm RSA jest systemem z kluczem publicznym i jego bezpieczeństwo opiera się na trudnym z punktu widzenia złożoności obliczeniowej algorytmie faktoryzacji wielkich liczb całkowitych N na iloczyn liczb pierwszych [3]. Dotychczas nie jest znany deterministyczny lub losowy algorytm o złożoności P-T (polynomial-time) tzn. o złożoności $O((\log N)^c)$ gdzie c jest pewną stałą. Stąd zrealizowany fizycznie kwantowy algorytm Shora mógłby zmienić warunki i poziom bezpieczeństwa niektórych systemów kryptograficznych. Niezależnie od oceny realizowalności uniwersalnych komputerów kwantowych, problem faktoryzacji wielkich liczb całkowitych jest aktualnie rozwiązywany także metodami teorii liczb [4]. Celem tego typu badań jest w szczególności określenie granic bezpieczeństwa systemów kryptograficznych. Warto zauważyć, że ostatnio opublikowano rezultat faktoryzacji liczby 291311 przy użyciu, według autorów, fizycznego obiektu kwantowego [5]. Z punktu widzenia systemu RSA jest to oczywiście bardzo mała liczba całkowita. Celem niniejszej publikacji jest wskazanie na możliwość implementacji niektórych z wyżej wymienionych algorytmów kwantowych wykorzystując struktury hamiltonowskich sieci neuronowych. Sieci takie są pewnym rozszerzeniem struktur sieci Hopfield'a, przy czym ich punkty równowagi prowadzą do sformułowania dwóch istotnych algorytmów:

1. sformułowanie struktur modułów oktonionicznych i najlepiej dopasowanych baz
2. sformułowanie transformacji ortogonalnych i biortogonalnych.

Algorytmy te są podstawą dla projektów modeli uczenia maszynowego [6, 7].

Hamiltonowskie sieci neuronowe i modele uczenia maszynowego

Fizyczne układy hamiltonowskie opisane są przez następujące równanie stanu:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\nabla H(\boldsymbol{x}) \tag{1}$$

gdzie: $\boldsymbol{x}$- wektor stanu, $\boldsymbol{x} \in \mathcal{R}^{2n}$; $\boldsymbol{J}$- macierz ortogonalna, skośnie symetryczna; $\nabla H(\boldsymbol{x})$- *gradient całkowitej energii* $H(\boldsymbol{x})$

Równanie (1) daje podstawy do sformułowania struktur sieci neuronowych, będących pewnym rozszerzeniem struktur sieci Hopfield'a, a mianowicie:

$$\dot{\boldsymbol{x}} = (\boldsymbol{W} - w_0\mathbf{1} + \varepsilon\boldsymbol{W}_s)\boldsymbol{\theta}(\boldsymbol{x}) + \boldsymbol{d} \tag{2}$$

gdzie: $\boldsymbol{W}$- macierz wag połączeń, skośnie-symetryczna, ortogonalna $(\boldsymbol{W}^2 = -\mathbf{1}), dim\,\boldsymbol{W} = 2^n$, $\mathbf{1}$-macierz jednostkowa; $\boldsymbol{W}_s$- macierz symetryczna; $\boldsymbol{d}$- wektor danych wejściowych; $\boldsymbol{\theta}(\boldsymbol{x})$- wektor funkcji aktywacji spełniający:

$$\mu_1 \le \frac{\theta(x_i)}{x_i} \le \mu_2,\ \mu_1, \mu_2 \in (0, \infty)$$
$$w_0 \ge 0, \varepsilon \in \mathcal{R}$$

Nietrudno zauważyć, że sieci neuronowe opisane powyższym równaniem posiadają dla $\varepsilon = 0$ punkty równowagi o właściwościach transformacji ortogonalnej:

$$\boldsymbol{y} = \theta(\boldsymbol{x}) = \frac{1}{1+w_0^2}(\boldsymbol{W} + w_0\mathbf{1})\boldsymbol{d} \tag{3}$$

gdzie wektor $\boldsymbol{y}$ reprezentuje widmo Haar'a wektora danych $\boldsymbol{d}$. W szczególności dla $dim\,\boldsymbol{W} = 8$ transformacja (3) definiuje tzw. moduł oktonioniczny $\boldsymbol{W}_0$:

$$\boldsymbol{y} = \theta(\boldsymbol{x}) = \frac{1}{1+w_0^2}(\boldsymbol{W}_8 + w_0\mathbf{1})\boldsymbol{d} = \boldsymbol{W}_0\boldsymbol{d} \tag{4}$$

gdzie:

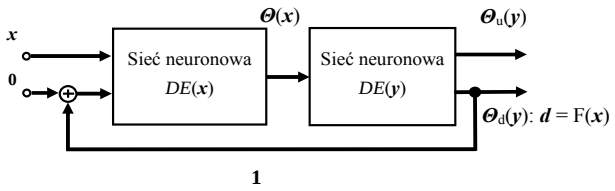
(5)
$$\boldsymbol{W}_8 = \begin{bmatrix} 0 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 \\ -w_1 & 0 & w_3 & -w_2 & w_5 & -w_4 & -w_7 & w_6 \\ -w_2 & -w_3 & 0 & w_1 & w_6 & w_7 & -w_4 & -w_5 \\ -w_3 & w_2 & -w_1 & 0 & w_7 & -w_6 & w_5 & -w_4 \\ -w_4 & -w_5 & -w_6 & -w_7 & 0 & w_1 & w_2 & w_3 \\ -w_5 & w_4 & -w_7 & w_6 & -w_1 & 0 & -w_3 & w_2 \\ -w_6 & w_7 & w_4 & -w_5 & -w_2 & w_3 & 0 & -w_1 \\ -w_7 & -w_6 & w_5 & w_4 & -w_3 & -w_2 & w_1 & 0 \end{bmatrix}$$

Kolumny i wiersze macierzy $(\mathbf{W}_8 + \mathrm{w}_0\mathbf{1})$ tworzą najlepiej dopasowaną bazę ortogonalną dla rozwiązania zadania transformacji zadanego wektora $\boldsymbol{d}$ na zadany wektor $\boldsymbol{y}$:

(6)
$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \end{bmatrix} = \frac{1}{\sum_{\mathrm{i}=1}^{8} y_{\mathrm{i}}^2} \begin{bmatrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \\ -y_2 & y_1 & -y_4 & y_3 & -y_6 & y_5 & y_8 & -y_7 \\ -y_3 & y_4 & y_1 & -y_2 & -y_7 & -y_8 & y_5 & y_6 \\ -y_4 & -y_3 & y_2 & y_1 & -y_8 & y_7 & -y_6 & y_5 \\ -y_5 & y_6 & y_7 & y_8 & y_1 & -y_2 & -y_3 & -y_4 \\ -y_6 & -y_5 & y_8 & -y_7 & y_2 & y_1 & y_4 & -y_3 \\ -y_7 & -y_8 & -y_5 & y_6 & y_3 & -y_4 & y_1 & y_2 \\ -y_8 & y_7 & -y_6 & -y_5 & y_4 & y_3 & -y_2 & y_1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{bmatrix}$$

przy ograniczeniu $w_0 > 0$, tzn. $\boldsymbol{y}^T \cdot \boldsymbol{d} > 0$

Należy zauważyć, że macierz $\boldsymbol{W}_8$ należy do klasy macierzy Hurwitza-Radona. Łącząc kompatybilne macierze $\boldsymbol{W}_8$ można uzyskać macierz wag $\boldsymbol{W}$ dla $dim\, \boldsymbol{W} = 2^n$, $n = 4, 5, \ldots$. Jak pokazano w publikacjach [6, 7] punkty równowagi sieci neuronowej (2) mogą determinować pewną transformację biortogonalną. Łącząc bowiem dwie sieci neuronowe dane równaniem (2) w pierścień, uzyskuje się ogólny model uczenia maszynowego rozwiązujący problem aproksymacji odwzorowania $\boldsymbol{d} = F(\boldsymbol{x})$ zadanego przez zbiór treningowy $S = \{\boldsymbol{x}_i, \boldsymbol{d}_i\}_{i=1}^N$. Połączenie takie pokazano na rysunku 1.



Rys. 1. Struktura modelu uczenia maszynowego oparta na sieciach neuronowych

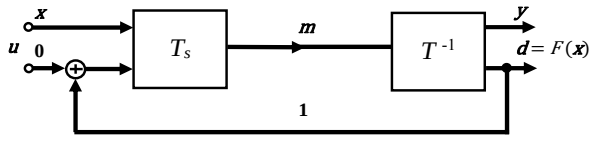
gdzie:

$$DE(\boldsymbol{x})\colon \dot{\boldsymbol{x}} = (\boldsymbol{W} - 2 \cdot \mathbf{1} + \boldsymbol{W}_s)\boldsymbol{\theta}(\boldsymbol{x}) + \begin{bmatrix} \boldsymbol{\Sigma} \\ \cdots \\ \mathbf{0} + \boldsymbol{\theta}_d(\boldsymbol{y}) \end{bmatrix}$$

(7)
$$DE(\boldsymbol{y})\colon \dot{\boldsymbol{y}} = \tfrac{1}{2}(-\boldsymbol{W} - \mathbf{1}) \begin{bmatrix} \boldsymbol{\theta}_u(\boldsymbol{y}) \\ \boldsymbol{\theta}_d(\boldsymbol{y}) \end{bmatrix} + \boldsymbol{\theta}(\boldsymbol{x})$$

Punkty równowagi modelu z rysunku 1 można zinterpretować przy pomocy schematu blokowego przedstawionego na rysunku 2 i wykonującego zaprojektowane działania algebraiczne: $\mathbf{d} = \mathrm{F}(\mathbf{x})$,

gdzie: $T_s(\cdot) = (2 \cdot \mathbf{1} - \boldsymbol{W}_s - \boldsymbol{W})^{-1}$
$T^{-1}(\cdot) = (-\boldsymbol{W} + \mathbf{1})$



Rys. 2. Schemat blokowy realizujący zaprojektowane działanie algebraiczne

Szczegóły projektu odwzorowania $\mathbf{d} = \mathrm{F}(\mathbf{x})$ można znaleźć w publikacji [6, 7].

Procesor analogowy

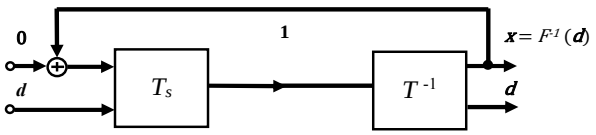
Sieć neuronową z rysunku 1 lub strukturę blokową z rysunku 2 można wykorzystać do realizacji procesora analogowego wykonującego operację dodawania komponentowego dwóch wektorów $\boldsymbol{d}_1$ i $\boldsymbol{d}_2$. Rzeczywiście zakładając, że wektor wejściowy

(8)
$$\boldsymbol{x} = \boldsymbol{x}_1 + \boldsymbol{x}_2$$

gdzie: $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$ oraz $\boldsymbol{x}$ jest znane, ponieważ odwzorowanie $\mathrm{F}(\cdot)$ spełnia zasadę superpozycji:

(9)
$$F\begin{bmatrix} \boldsymbol{x}_1 \\ \cdots \\ \mathbf{0} \end{bmatrix} \rightarrow \begin{bmatrix} \boldsymbol{x}_1 \\ \cdots \\ \boldsymbol{d}_1 \end{bmatrix}, \qquad F\begin{bmatrix} \boldsymbol{x}_2 \\ \cdots \\ \mathbf{0} \end{bmatrix} \rightarrow \begin{bmatrix} \boldsymbol{x}_2 \\ \cdots \\ \boldsymbol{d}_2 \end{bmatrix}$$
$$\text{więc: } F\begin{bmatrix} \boldsymbol{x} \\ \cdots \\ \mathbf{0} \end{bmatrix} = F\begin{bmatrix} \boldsymbol{x}_1 + \boldsymbol{x}_2 \\ \cdots \\ \mathbf{0} \end{bmatrix} \rightarrow \begin{bmatrix} \boldsymbol{y} = \boldsymbol{x}_1 + \boldsymbol{x}_2 \\ \cdots \\ \boldsymbol{d}_1 + \boldsymbol{d}_2 \end{bmatrix}$$

Działanie takiego procesora opiera się na podstawowej własności systemu, a mianowicie na zdolności do rekonstrukcji danych (tutaj $\boldsymbol{d}_1$ oraz $\boldsymbol{d}_2$). Strukturę procesora analogowego można wykorzystać do operacji mnożenia komponentowego zadanego wektora $\boldsymbol{x}$ przez stałą $C \in \mathcal{R}$ tzn. $C \cdot \boldsymbol{x}$. Zmieniając bowiem strukturę pętli, jak pokazano na rysunku 3, realizuje się model odwzorowania odwrotnego $\boldsymbol{x} = F^{-1}(\boldsymbol{d})$.



Rys. 3. Schemat blokowy procesora realizującego model odwzorowania odwrotnego

Tak więc wyżej wymieniona operacja mnożenia dana jest przez odwzorowania:

(10)
$$F:\ \begin{bmatrix} \boldsymbol{x} \\ \cdots \\ \mathbf{0} \end{bmatrix} \rightarrow \begin{bmatrix} \boldsymbol{x} \\ \cdots \\ \mathbf{1}_w \end{bmatrix},\ F^{-1}\begin{bmatrix} \mathbf{1}_w \\ \cdots \\ \boldsymbol{x} \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{1}_w \\ \cdots \\ \boldsymbol{x} \end{bmatrix} \rightarrow \begin{bmatrix} C \cdot \mathbf{1}_w \\ \cdots \\ C \cdot \boldsymbol{x} \end{bmatrix}$$

gdzie: $\mathbf{1}_w = [1, \ldots, 1]^T$.

Sposób wykorzystania procesora analogowego realizującego odwzorowanie (10) przytoczono poniżej w przykładzie faktoryzacji liczb całkowitych.

Faktoryzacja liczb całkowitych

Prezentowane w pracy metody faktoryzacji liczb całkowitych *N* będących iloczynem dwóch liczb pierwszych, tzn:

(11)
$$\mathrm{N} = \mathrm{p} \cdot \mathrm{q}$$

gdzie: *p*, *q* liczby pierwsze

oparta jest na założeniu, że moduły oktonioniczne oraz procesory analogowe mogą być zrealizowane jako obiekty fizyczne.

Faktoryzacja z wykorzystaniem modułów oktonionicznych

Niech $N = p \cdot q$, gdzie p, q- liczby pierwsze, więc:

(12)
$$p \text{ lub } q \leq N^{1/2} \text{ oraz } N \cdot \tfrac{1}{p} = q$$

Stąd ciąg liczb pierwszych $\{p_i\}_{i=1}^{L}$ można traktować jako zbiór wektorów 8-elementowych $\boldsymbol{p}_k$, $k = 1,2,\dots,{}^{L}/_{8}$ gdzie:

(13) $$\boldsymbol{p}_1^T = [p_{11}, p_{12}, \dots, p_{18}],\ \boldsymbol{p}_2^T = [p_{21}, p_{22}, \dots, p_{28}], \dots,\ \boldsymbol{p}_k^T = [p_{k1}, p_{k2}, \dots, p_{k8}], \dots$$

L - ilość liczb pierwszych o wartościach $\leq \sim\sqrt{N}$, ($\sim\sqrt{N}$- liczba całkowita, zaokrąglenie $\sqrt{N}$) (liczba *L* może być także ustalona korzystając z funkcji $\pi(x)$ Eulera).

Dla każdego z wektorów $\boldsymbol{p}_k$ można wyznaczyć taką bazę ortogonalną($\mathbf{W}_8 + \mathrm{w}_0\mathbf{1}$), zgodnie z zależnością (5) i (6), aby otrzymać odpowiednio:

(14) $$\frac{1}{1+W_0^2}(\boldsymbol{W}_{8k} + w_0\mathbf{1}) \cdot \begin{bmatrix} \frac{1}{p_{k1}} \\ \vdots \\ \frac{1}{p_{k8}} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix},\ k = 1,2,\dots$$

Stąd:

(15) $$(\boldsymbol{W}_{8k}^T + w_0\mathbf{1}) \cdot \begin{bmatrix} N \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} q_{k1} \\ \vdots \\ q_{k8} \end{bmatrix} = \boldsymbol{q}_k,\ k = 1,2,\dots$$

Jeden z komponentów wektora $\boldsymbol{q}_k$ jest rozwiązaniem równania faktoryzacji (11). Systemowe rozwiązanie równania (15) wymaga równoległego połączenia $L/8$ modułów oktonionicznych (skrócenie czasu obliczeń przez równoległe połączenie procesorów).

Przykład 1

Dla zadanej liczby $N = 9991$ wyznacza się *L* = 24 gdyż $\sim\sqrt{N} = 100$. Tak więc do faktoryzacji liczby *N* potrzeba zrealizować 3 moduły oktonioniczne. W takim przypadku wektory $\boldsymbol{p}_k$ przyjmują postać:

$\boldsymbol{p}_1^T = \left[\frac{1}{3}, \frac{1}{5}, \frac{1}{7}, \frac{1}{11}, \frac{1}{13}, \frac{1}{17}, \frac{1}{19}, \frac{1}{23}\right]$,
$\boldsymbol{p}_2^T = \left[\frac{1}{29}, \frac{1}{31}, \frac{1}{37}, \frac{1}{41}, \frac{1}{43}, \frac{1}{47}, \frac{1}{53}, \frac{1}{59}\right]$,
$\boldsymbol{p}_3^T = \left[\frac{1}{61}, \frac{1}{67}, \frac{1}{71}, \frac{1}{73}, \frac{1}{79}, \frac{1}{83}, \frac{1}{89}, \frac{1}{97}\right]$

są zatem transformowane zgodnie z zależnością (15) do postaci:

$\boldsymbol{q}_1^T = [3330{,}30\ 1998{,}20\ \ 1427{,}29\ 908{,}27\ 768{,}54\ 587{,}71\ 525{,}84\ 434{,}39]$,

$\boldsymbol{q}_2^T = [344{,}51\ 322{,}29\ 270{,}02\ 243{,}68\ 232{,}34\ 212{,}57\ 188{,}50\ 169{,}33]$,

$\boldsymbol{q}_3^T = [163{,}78\ 149{,}11\ 140{,}71\ 136{,}86\ 126{,}46\ 120{,}37\ 112{,}25\ 103{,}00]$.

Element o wartości 103 wektora $\boldsymbol{q}_3$ jest poszukiwaną liczbą pierwszą: 9991/103=97.

Przykład 2

Faktoryzacja liczby $N = 302342543$, jakkolwiek relatywnie większej niż 9991, dokonywana jest według tej samej procedury. Tak więc: $\sim\sqrt{N} = 17388$ oraz $L \leq 2000/8$. Zatem dla faktoryzacji liczby *N* należy użyć maksymalnie 250 modułów oktonionicznych. W takim przypadku wektor:

$$\boldsymbol{p}_{250}^T = \left[\frac{1}{17333}, \frac{1}{17341}, \frac{1}{17351}, \frac{1}{17359}, \frac{1}{17377}, \frac{1}{17383}, \frac{1}{17387}, \frac{1}{17389}\right]$$

transformowany jest zgodnie z zależnością (14) do postaci:

$\boldsymbol{q}_{250}^T$ = [17443,1744648936 17435,1273282971 17425,0788427180 17417,0483898842 17399,0069056799 17393,0013806593 17389,0000000000 17387,0000000000]

Elementy o wartościach 17389 oraz 17387 są poszukiwanymi liczbami pierwszymi: $302342543 = 17389 \cdot 17387$.

Faktoryzacja z wykorzystaniem procesora analogowego

Przykład 3

Możliwość faktoryzacji liczby *N*=*p*·*q*, wykorzystując dodawanie komponentowe wektorów przez procesor analogowy przedstawiony na rysunku 1 lub rysunku 2, dana jest przez logarytmowanie zależności (11):

(16) $$N \cdot \frac{1}{p} = q;\quad \ln N - \ln p = \ln q$$

Tak więc odwzorowania (9) mają postać:

(17) $$F\begin{bmatrix} \boldsymbol{x}_1 \\ \cdots \\ \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \boldsymbol{x}_1 \\ \cdots \\ \ln N \\ \vdots \\ \ln N \end{bmatrix},\ F\begin{bmatrix} \boldsymbol{x}_2 \\ \cdots \\ \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \boldsymbol{x}_2 \\ \cdots \\ \ln p_1 \\ \vdots \\ \ln p_L \end{bmatrix}$$

zatem $$F\begin{bmatrix} \boldsymbol{x}_1 - \boldsymbol{x}_2 \\ \cdots \\ \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \boldsymbol{x}_1 - \boldsymbol{x}_2 \\ \cdots \\ \ln N - \ln p_1 \\ \vdots \\ \ln N - \ln p_L \end{bmatrix}$$

Tylko jeden z liczbowych rezultatów: $e^{(\ln N - \ln p_i)}, i \in \{1, \dots, L\}$ będący liczbą całkowitą jest rozwiązaniem zadania faktoryzacji liczby *N*. Realizowalność procesora analogowego wymaga spełnienia warunku: $\dim \boldsymbol{x}_i + L = 2^n$. Zaprojektowanie procesora wykonującego zależność (17) jest obliczeniowo stabilne, ale taka metoda faktoryzacji ze względu na użycie funkcji nieliniowych nie może być wykorzystywana dla wielkich liczb *N*.

Przykład 4

Faktoryzacja liczby 302342543, za pomocą procesora mnożącego z rysunku 3 została zrealizowana w przestrzeni $\dim \boldsymbol{W} = 2^{12} = 4096$. Tak więc:

(18) $$F:\ \begin{bmatrix} \boldsymbol{x} \\ \cdots \\ \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \boldsymbol{x} \\ \cdots \\ \mathbf{1}_w \end{bmatrix} = \begin{bmatrix} \boldsymbol{p} \\ \cdots \\ \mathbf{1}_w \end{bmatrix}$$

gdzie: $\boldsymbol{x} = \boldsymbol{p} = \left[\frac{1}{p_1}, \frac{1}{p_2}, \dots, \frac{1}{p_L}\right]^T$; $L = 2048, p_i$- liczby pierwsze *i* = 1, 2, … ,2048

$\mathbf{1}_w = [1, \dots, 1]^T, \dim \mathbf{1}_w = 2048$

(19) $$F^{-1}:\ \begin{bmatrix} \mathbf{1}_w \\ \cdots \\ \mathbf{0} \end{bmatrix} \to \begin{bmatrix} \mathbf{1}_w \\ \cdots \\ \boldsymbol{p} \end{bmatrix} \to \begin{bmatrix} N \cdot \mathbf{1}_w \\ \cdots \\ N \cdot \boldsymbol{p} \end{bmatrix}$$

Jeden ze składników wektora $N \cdot \boldsymbol{p}$ jest poszukiwaną liczbą pierwszą.

Podsumowanie

Możliwość zastosowania modeli uczenia maszynowego do rozwiazywania zagadnień sformułowanych jako algorytmy kwantowe wskazuje na ich uniwersalność. Szczególnie intersującym tutaj zagadnieniem jest faktoryzacja liczb całkowitych. Wydaje się, że złamanie systemu kryptograficznego RSA (np. 1024(bity)-RSA) w czasie o złożoności P-T jest możliwe jedynie przez zastosowanie specjalizowanych procesorów takich jak komputer kwantowy (algorytm Shor'a). W niniejszej pracy wskazano jednakże na model faktoryzacji oparty na transformacjach ortogonalnych nazwanych modułami oktonicznymi, których użycie, po ich realizacji fizycznej i technologicznej VLSI, mogłoby stanowić narzędzie dla w/w faktoryzacji. Warto zauważyć, że także inne algorytmy kwantowe (np. Deutsch'a–Jozsa) pozwalające na przyśpieszenie przeszukiwania baz danych (np. problem „oracle") mogą być efektywnie emulowane korzystając z własności opisanego procesora analogowego.

***Autorzy**: dr inż. Wiesław Citko, Uniwersytet Morski w Gdyni, Wydział Elektryczny, ul. Morska 81-87, 81-225 Gdynia; E-mail: w.citko@we.umg.edu.pl; dr hab. inż. Wiesław Sieńko, Uniwersytet Morski w Gdyni, Wydział Elektryczny, ul. Morska 81-87, 81-225 Gdynia, E-mail: w.sienko@we.umg.edu.pl.*

LITERATURA

[1] Preskill J., Quantum information and computation, *Lecture Note of Physics 229*, California Institute of Technology, Los Angeles, (1998).

[2] Shor P., Algorithms for quantum computation: discreet logarithms and factoring, *Proc. 35th Annually Symposium on Foundations of Computer Science*, IEEE Computer Society Press, Los Alamitos,(1994), 124-134.

[3] Rivest R. L., Shamir A., Adleman L., A method of obtaining digital signatures and public-key cryptosystems, *Comm. ACM 21*, (1978), 120-126.

[4] Brent R. P., Recent progress and prospects for integer factorisation algorithms, *International Computing and Combinatorics Conference,* COCOON 2000, LNCS 1858, (2000), 3-22.

[5] Zhaokai Li, Dattani N. S. i inni, High-fidelity adiabatic quantum computation using the intrinsic Hamiltonian of spin system: Application to the experimental factorization of 291311, (2017), *arXiv: 1706.0806v1*.

[6] Citko W., Sieńko W., Realizacja pamięci skojarzeniowej z zastosowaniem uczenia maszynowego, *Przegląd Elektrotechniczny*, (2017), n.8, 77-80.

[7] Citko W. , Sieńko W., Zastosowaniem uczenia maszynowego do realizacji procesora analogowego, *Przegląd Elektrotechniczny*, (2018), n.9, 56-58.