RBI Quantum Hackathon Workbench
- this time on Learning Parity with Noise from Machine Learning Viewpoint

Tomas Rosa* and Jiri Pavlu, CBCC of RBI in Prague

Bratislava, October, 2019

*) corresponding author

Revision History

- 06/10/2019/Tom - LPN introduction, Nature-style experiment elaboration
- 07/10/2019/Tom - Generalised ancilla initialisation leading to a better algorithm
- 08/10/2019/Tom - Improved readability by detailing key formulas derivations
- 17/10/2019/Tom - Epsilon ancilla qubit incorporated

Notation Notes

- $\oplus$ denotes (vector) addition modulo 2
- $\odot$ denotes (vector) multiplication modulo 2
    - for binary vectors, $\odot$ is a standard dot product modulo 2
    - when clear from the context, we use simply $+$ and $\oplus$, or $\cdot$ and $\odot$ interchangeably

Symmetrisation Intermezzo

- We will often work with output of boolean functions like $f(x)$, where x is a binary vector
- This output is in general either 0 or 1
- When designing quantum algorithms, we need to incorporate $f(x)$ into superposition coefficients in a concise way to see the effect of quantum operators
- For this, it is useful to transform $f(x)$ so to make its result either -1 or 1, or to incorporate the effect of $f(x)$ through a power of (-1)

Symmetrisation We Use

Let $f(x) \in \{0,1\}$.

Then:

$$2f(x)-1 \in \{-1,1\},\ -1 \text{ iff } f(x)=0$$

$$(-1)^{f(x)} \in \{-1,1\},\ -1 \text{ iff } f(x)=1$$

In particular:

$$(-1)^{f(x)} = 1-2f(x)$$

$$1+(-1)^{f(x)} = 2(1-f(x))$$

$$1-(-1)^{f(x)} = 2f(x)$$

Also note the Hadamard transform of such a boolean *f*(*x*)

$$\left|f(x)\right\rangle \mapsto \frac{(-1)^{0\cdot f(x)}\left|0\right\rangle + (-1)^{1\cdot f(x)}\left|1\right\rangle}{\sqrt{2}}$$

$$= \frac{\left|0\right\rangle + (-1)^{f(x)}\left|1\right\rangle}{\sqrt{2}}$$

Learning Parity with Noise

The *search* version of the learning parity with noise problem with parameters $\ell \in \mathbb{N}$ (the length of the secret), $\tau \in \mathbb{R}$ where $0 < \tau < 0.5$ (the noise rate) and $q \in \mathbb{N}$ (the numbers of samples) asks to find a fixed random ℓ bit secret $\mathbf{s} \in \mathbb{Z}_2^\ell$ from q samples of the form $\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle \oplus e$ where $\mathbf{a} \in \mathbb{Z}_2^\ell$ is random and $e \in \mathbb{Z}_2$ has Bernoulli distribution with parameter τ (we denote this distribution with Ber_τ), i.e. $\Pr[e = 1] = \tau$. The *decisional* LPN problem is defined similarly, except that we require that one cannot even distinguish noisy inner products from random.

[Krzysztof Pietrzak, 2012]

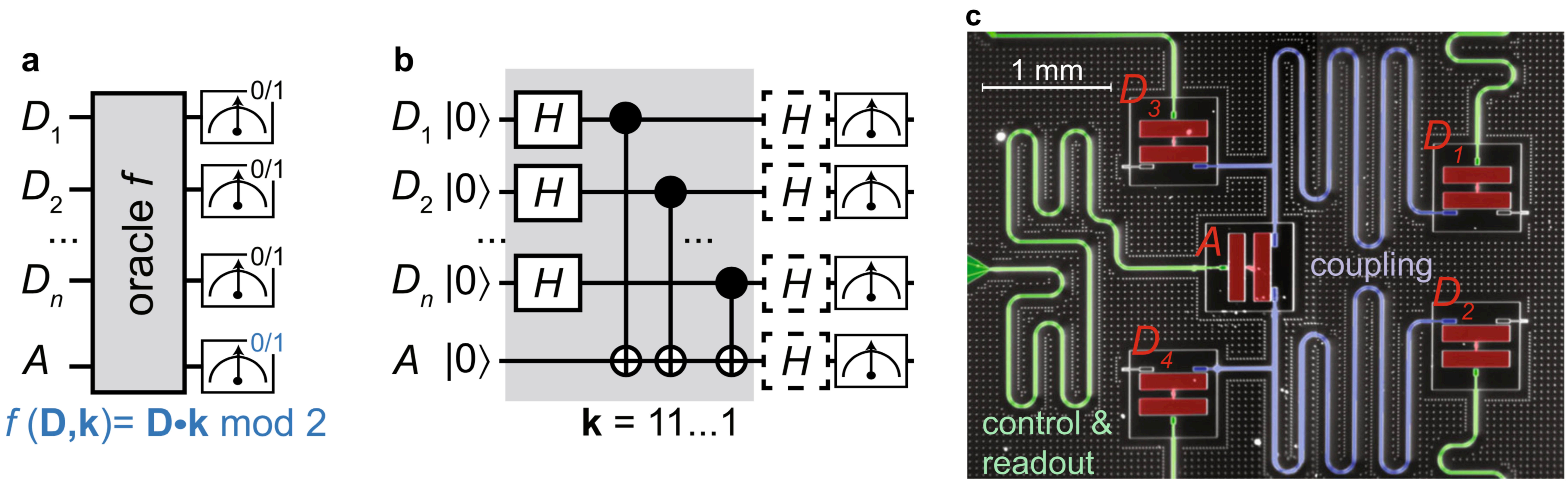
Broad Impact of LPN

- We have chosen the Learning Parity with Noise problem for this hackathon, since it has considerable impact on both

  - machine learning techniques

  - post-quantum cryptography and cryptanalysis

Starting Experiment Described in Nature Partner Journal on QI (4/17)



ARTICLE OPEN

Demonstration of quantum advantage in machine learning

Diego Ristè[1], Marcus P. da Silva[1], Colm A. Ryan[1], Andrew W. Cross[2], Antonio D. Córcoles[2], John A. Smolin[2], Jay M. Gambetta[2], Jerry M. Chow[2] and Blake R. Johnson[1]

Our Starting Position

- We consider the quantum approach to LPN solving, so we always assume the final Hadamard gates are on
  - omitting output Hadamard(s) was to simulate classical LPN conditions with the same QPU (*Quantum Processing Unit*) core setup
- In the original, the LPN noise is intrinsic, generated by QPU inherently
  - we stick more with classical LPN, so we explicitly use the additive error factor
  - it is generated independently for each oracle-operator invocation

Our Goals

- Improve the efficiency of the original algorithm by showing its direct connection with Bernstein-Vazirani algorithm we elaborated in Vienna in May this year (so btw., there is an ongoing competence extension and application)
  - it follows from a generalisation of the ancilla qubit initialisation
  - actually, it is a bit surprising the former authors did not note this connection already
- Verify the theoretical construction practically, even with a higher number of data qubits (originally, they used two)
  - we want to show this approach practically halves the number of LPN oracle invocations (respectively the number of QPU runs); as we are talking about practical machine learning algorithms, this can be significant
- Discuss the quantum advantage
  - we should be able to solve even the worst-case LPN instances that are unsolvable classically
  - i.e. for $\tau = 1/2$

We investigate the original ("Nature-style") initialisation and approach

$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in F_2^{\,n}} |x\rangle \otimes |0\rangle$$

note we have dropped the bar vector notation used before

We apply the LPN oracle operator for $f_k(x)$
k is the hidden number, $\boldsymbol{\varepsilon}$ is the binary error factor (the noise)

$$|x\rangle \otimes |0\rangle \mapsto |x\rangle \otimes |f_i(x) \oplus 0\rangle, \text{ where } f_k(x) = k \odot x \oplus \varepsilon$$

$$|\psi_{2,k}\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in F_2{}^n} |x\rangle \otimes \left(\frac{1+(-1)^{f_k(x)}}{2} |0\rangle + \frac{1-(-1)^{f_k(x)}}{2} |1\rangle \right)$$

phase kickback effect **not obvious**, now

Then we apply the final Hadamard transform(s) (cf. original scheme)

$$\left|\psi_{3,k}\right\rangle = \frac{1}{2^n}\sum_{y\in F_2{}^n}\sum_{x\in F_2{}^n}(-1)^{x\odot y}\left|y\right\rangle\otimes\left[\frac{1+(-1)^{f_k(x)}}{2\sqrt{2}}\left(\left|0\right\rangle+\left|1\right\rangle\right)+\frac{1-(-1)^{f_k(x)}}{2\sqrt{2}}\left(\left|0\right\rangle-\left|1\right\rangle\right)\right]$$

$$=\frac{1}{2^n}\sum_{y\in F_2{}^n}\sum_{x\in F_2{}^n}(-1)^{x\odot y}\left|y\right\rangle\otimes\left(\frac{\left|0\right\rangle+(-1)^{f_k(x)}\left|1\right\rangle}{\sqrt{2}}\right),\ \text{for } f_k(x)=k\odot x\oplus\varepsilon$$

$$=\frac{1}{2^n\sqrt{2}}\sum_{y\in F_2{}^n}\sum_{x\in F_2{}^n}(-1)^{x\odot y}\left|y\right\rangle\left|0\right\rangle+\frac{1}{2^n\sqrt{2}}\sum_{y\in F_2{}^n}\sum_{x\in F_2{}^n}(-1)^{\varepsilon}(-1)^{x\odot(y\oplus k)}\left|y\right\rangle\left|1\right\rangle$$

$$=\frac{1}{\sqrt{2}}\underbrace{\left|00\ldots0\right\rangle}_{\text{garbage}}\left|0\right\rangle+\frac{(-1)^{\varepsilon}}{\sqrt{2}}\underbrace{\left|k\right\rangle}_{\text{direct secret bits}}\left|1\right\rangle$$

Notes on the Original Approach

- We have 50% chance to measure |0> and |1> on the ancilla qubit, respectively
  - measuring |0> brings no further information in data qubits
  - measuring |1> reveals the hidden number (secret) k in data qubits
- In case of the positive answer, the result is totally insensitive on the ε noise
  - this is certainly a good point (though not so much addressed before)
- We, however, waste around 50% of QPU runs
  - we shall try to do this better, now

In Search for the Generalised Ancilla Initialisation (GAI)

- Note that the original “Nature-style” approach works practically the same way, regardless whether the ancilla is initialised as |0> or |1>
  - this suggests that both of the pure eigenstates are equally good or bad
  - how about to try their superposition?
  - heuristically, we try an equal superposition with a variable relative phase

The Generalised Ancilla Initialisation (GAI)

$$\left|\psi_1\right\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in F_2{}^n} \left|x\right\rangle \otimes \left(\frac{\left|0\right\rangle + e^{i\varphi}\left|1\right\rangle}{\sqrt{2}} \right)$$

Again, we apply the LPN oracle operator for $f_k(x)$

$$\left|\psi_{2,k}\right\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in F_2^{\,n}} \left|x\right\rangle \otimes \left(\frac{\left|f_k(x)\right\rangle + e^{i\varphi}\left|f_k(x) \oplus 1\right\rangle}{\sqrt{2}} \right)$$

phase kickback can be shown for e.g. $\boldsymbol{\varphi} = \boldsymbol{\pi}$

Now, we use the final Hadamard transform(s) to see how it works

$$\left|\psi_{3,k}\right\rangle = \frac{1}{2^n}\sum_{y\in F_2^{\ n}}\sum_{x\in F_2^{\ n}}(-1)^{x\odot y}\left|y\right\rangle\otimes\left(\frac{\left|0\right\rangle+(-1)^{f_k(x)}\left|1\right\rangle}{2}+e^{i\varphi}\frac{\left|0\right\rangle+(-1)^{f_k(x)\oplus 1}\left|1\right\rangle}{2}\right)$$

$$=\frac{1}{2^n}\sum_{y\in F_2^{\ n}}\sum_{x\in F_2^{\ n}}(-1)^{x\odot y}\left|y\right\rangle\otimes\left(\frac{(1+e^{i\varphi})}{2}\left|0\right\rangle+\frac{(1-e^{i\varphi})(-1)^{f_k(x)}}{2}\left|1\right\rangle\right),\ \text{note } (-1)^{f_k(x)\oplus 1}=-(-1)^{f_k(x)}$$

$$=\frac{1+e^{i\varphi}}{2^{n+1}}\sum_{y\in F_2^{\ n}}\sum_{x\in F_2^{\ n}}(-1)^{x\odot y}\left|y\right\rangle\left|0\right\rangle+\frac{1-e^{i\varphi}}{2^{n+1}}\sum_{y\in F_2^{\ n}}\sum_{x\in F_2^{\ n}}(-1)^{\varepsilon}(-1)^{x\odot(y\oplus k)}\left|y\right\rangle\left|1\right\rangle,\ \text{note } f_k(x)=k\odot x\oplus\varepsilon$$

$$=\frac{1+e^{i\varphi}}{2}\underbrace{\left|00\ldots0\right\rangle}_{\text{garbage}}\left|0\right\rangle+\frac{(-1)^{\varepsilon}(1-e^{i\varphi})}{2}\underbrace{\left|k\right\rangle}_{\text{direct secret bits}}\left|1\right\rangle$$

Minimising the Garbage Probability

- Using GAI, we can manipulate the probabilty of the garbage |00…0>|0> state observation
  - this way, we can minimise the waste of QPU runs, so to increase effectiveness of our LPN quantum solver
- The garbage probability is equal to zero if

$$1 + e^{i\varphi} = 0 \Leftrightarrow \varphi = (2b+1)\pi,\ b \in \mathbb{Z}$$

Welcome back, please, Mr. Bernstein and Mr. Vazirani

- With $\boldsymbol{\varphi} = \boldsymbol{\pi}$, we get exactly the Bernstein-Vazirani algorithm (BVA) again
  - as reformulated by Cleve et al. in “*Quantum Algorithms Revisited*”, 1997
  - apparently, this algorithm is quite powerful for both machine learning and cryptology
  - **we have shown that BVA is a general extension of the Nature-style approach and the most efficient way to solve the LPN studied here**

Experimental Implementation of the ε-error

- To fully implement the LPN oracle, we would need to be able to alter its quantum operator for each and every QPU run
  - hard to do with the actual Qiskit platform
- We decided to do an equivalent implementation based on an extra error-driving qubit
  - the error is still interpreted classically, but it is inserted in a quantum way
  - adding the $|q_\varepsilon\rangle$ ancilla qubit tweaks the state after the "error-free" LPN operator using a CNOT entanglement to a superposition of error-free and erroneous substates
  - our LPN solver solves both instances in parallel; finally revealing only one, depending on the error probability distribution
  - by measuring the epsilon ancilla, we can get a kind of "debug" information for further statistical processing

Epsilon Ancilla Qubit Before Entangling with the LPN Output (General Distribution)

$$\left|\psi_{2,k}\right\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in F_2{}^n} \left|x\right\rangle \otimes \left(\frac{\left|f_k(x)\right\rangle + e^{i\varphi}\left|f_k(x)\oplus 1\right\rangle}{\sqrt{2}} \right) \underbrace{\otimes}_{\text{epsilon ancilla added; state is tweaked via tensor product}} \left(\cos\frac{\theta}{2}\left|0_\varepsilon\right\rangle + \sin\frac{\theta}{2} e^{i\beta}\left|1_\varepsilon\right\rangle \right)$$

— here, the f_k is just the inner product, without the epsilon error

CNOT Entangling the Epsilon Error with the LPN Output (General Error Distribution)

$$\left|\psi^{(\varepsilon)}{}_{2,k}\right\rangle = \frac{\cos\frac{\theta}{2}}{\sqrt{2^n}} \sum_{x\in F_2{}^n} \left|x\right\rangle \otimes \left(\frac{\left|f_{k,\varepsilon=0}(x)\right\rangle + e^{i\varphi}\left|f_{k,\varepsilon=0}(x)\oplus 1\right\rangle}{\sqrt{2}} \right) \otimes \left|0_\varepsilon\right\rangle$$

— error-free branch

$$+\frac{\sin\frac{\theta}{2}e^{i\beta}}{\sqrt{2^n}} \sum_{x\in F_2{}^n} \left|x\right\rangle \otimes \left(\frac{\left|f_{k,\varepsilon=1}(x)\right\rangle + e^{i\varphi}\left|f_{k,\varepsilon=1}(x)\oplus 1\right\rangle}{\sqrt{2}} \right) \otimes \left|1_\varepsilon\right\rangle$$

— erroneous branch

— here, the $f_{k,\varepsilon}$ is the inner product with the explicit epsilon error

What Follows

- Standard finalisation via $|\psi_3>$
  - both error-free and erroneous branches are solved in parallel
  - the measurement finally reveals either the branch for ε=0 or ε=1, respectively
  - by observing the epsilon ancilla qubit, we can get further statistical discrimination to verify our solver works for both situations equally well