EPJ Web of Conferences **245**, 10006 (2020) https://doi.org/10.1051/epjconf/202024510006
CHEP 2019

Quantum annealing algorithms for track pattern recognition

Masahiko Saito[1,*], *Paolo* Calafiura[2], *Heather* Gray[2,3], *Wim* Lavrijsen[2], *Lucy* Linder[2,4], *Yasuyuki* Okumura[1], *Ryu* Sawada[1], *Alex* Smith[3], *Junichi* Tanaka[1], and *Koji* Terashi[1]

[1]International Center for Elementary Particle Physics (ICEPP), The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-0033, Japan
[2]Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[3]Department of Physics, University of California, Berkeley, CA 94720, USA
[4]Haute ecole d'Ingénierie et d'Architecture de Fribourg, Boulevard de Pérolles 80, 1705 Fribourg, Switzerland

Abstract. The High-Luminosity Large Hadron Collider (HL-LHC) starts from 2027 to extend the physics discovery potential at the energy frontier. The HL-LHC produces experimental data with a much higher luminosity, requiring a large amount of computing resources mainly due to the complexity of a track pattern recognition algorithm. Quantum annealing might be a solution for an efficient track pattern recognition in the HL-LHC environment. We demonstrated to perform the track pattern recognition by using the D-Wave annealing machine and the Fujitsu Digital Annealer. The tracking efficiency and purity for the D-Wave quantum annealer are comparable with those for a classical simulated annealing at a low pileup condition, while a drop in performance is found at a high pileup condition, corresponding to the HL-LHC pileup environment. The tracking efficiency and purity for the Fujitsu Digital Annealer are nearly the same as the classical simulated annealing.

1 Introduction

After the discovery of a Higgs boson, the main targets in the energy frontier experiments at Large Hadron Collider (LHC) are a precise measurement of the Standard Model processes and the discovery of new physics phenomena beyond the Standard Model. To maximize the sensitivity for such phenomena, the LHC is planned to be upgraded from 2027, named High-Luminosity LHC (HL-LHC) [1]. The beam intensity of the HL-LHC increases several times higher than the LHC, resulting in high luminosity ($L = 5 \times 10^{34}\mathrm{cm}^{-2}\mathrm{s}^{-2}$), large number of collisions (pileup) per bunch crossing ($\langle\mu\rangle \sim 200$) and high readout rate in each subdetector. As well as stable detector operation, it is a challenging task to efficiently process the enormous amount of data in the physics analysis and object reconstruction step.

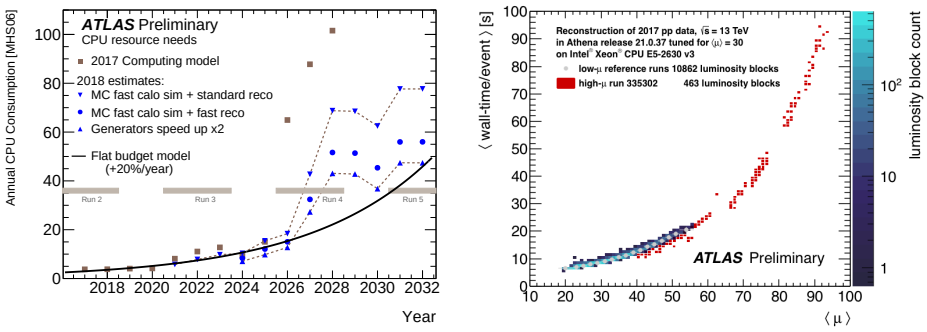
Track pattern recognition (*tracking*) is expected to seriously suffer from the increased pileup as shown in Figure 1 because the track reconstruction algorithm is based on a combinatorial algorithm. Next-generation collider experiments require a new idea of a track pattern recognition to reduce the computing time while keeping the tracking performance.

*e-mail: masahiko.saito@cern.ch

Recent novel techniques could solve the track pattern recognition task with a better performance. In the TrackML Particle Tracking Challenge [2] held at Kaggle and CodaLab, several winner algorithms [3] use a machine learning technique to improve the tracking performance. The HEP.TrkX [4] project also proposed new algorithms with learning methods based on deep neural networks. Furthermore, some approaches using a quantum computer are also proposed [5–8].

This paper focuses on an application of a quantum computer, which has recently become easy-to-access as some companies provide their devices for commercial use. It is known that some quantum algorithms have a clear advantage in solving complex computational problems, e.g., solving an inverse matrix task [9] and factorization of prime numbers [10]. Since a quantum computer uses quantum bits (qubits), which can represent a superposition of 0 and 1 states, the Hilbert space grows exponentially by increasing the number of qubits. A quantum computer is expected to have an advantage in solving a combinatorial optimization problem and would be a promising alternative for solving a track pattern recognition problem.

There are two types of quantum computing devices. One is a gate-based quantum computer, which uses quantum gates to solve the problem, and can be used for general-purpose tasks. The gate-based quantum computer is progressing fast, as demonstrated by Google recently claiming the achievement of quantum supremacy [11]. The number of qubits, however, is limited, e.g., 53 qubits for a Google quantum device [11] and IBM quantum device [12]. The other is an annealing-based quantum device, which solves problem by slowly changing the Hamiltonian and searching for the lowest energy state with a quantum fluctuation. It is not a general-purpose device, but is expected to solve the specific type of problem efficiently such as the minimization of Ising Hamiltonian. Based on the previous study [13], we demonstrate the algorithms to solve a track pattern recognition using an annealing technique with the D-Wave quantum annealer and the Fujitsu Digital Annealer.



(a) (b)

Figure 1: (a) Estimated CPU resources (b) Object reconstruction time as a function of pileup ($\langle\mu\rangle$) [14]

2 Methodology

2.1 Track pattern recognition

Track pattern recognition is to find the correct combination of sensors activated by a charged particle along the particle trajectory. The correct set of sensors is chosen by using spatial

positions of activated sensors, called *hits*. A trajectory of a charged particle can be modeled with five parameters. The main strategy of a good track pattern recognition is to check the consistency of hit positions on a possible trajectory. Since the number of hits in a trajectory is usually designed to be much more than the number of parameters of the trajectory, the correct hit set can be selected with rejecting combinatorial random coincidences.

This study uses the dataset prepared for the TrackML Particle Tracking Challenge [2], which was created assuming the HL-LHC pileup environment. The tracking detector consists of ten barrel layers and six or seven end-cap layers for both sides. The typical number of hits of a charged particle track in the barrel region is ten. The number of tracks is about ten thousand per bunch crossing, and the total number of hits is about one hundred thousand per bunch crossing, corresponding to the HL-LHC environment. In this study we focus on high transverse-momentum (p_{T}) tracks with p_{T} more than 1 GeV produced in the barrel region. The tracking performance is evaluated only for high p_{T} tracks, although all hits produced by low p_{T} tracks are used in the tracking.

2.2 Track parameters

A trajectory of a charged particle is characterized with five parameters. If a charged particle is not interacting with any fields or particles, the trajectory is a straight line, which can be modeled by four parameters: polar angle (θ), azimuthal angle (ϕ), a transverse distance from a collision point (d_0) and a longitudinal distance from a collision point (z_0). In a practical detector, there is a magnetic field along the beam axis to measure the transverse momentum, resulting in a charged particle curling and moving as a helix. The additional track parameter is a bending curvature, written as $1/R$ in this paper, where R is a Menger curvature.

2.3 Hamiltonian

A Hamiltonian for an annealing algorithm can be designed such that the ground state corresponds to the correct set of track patterns.

The previous study [15] using a Hopfield network for a track finding defines a Hamiltonian as follows

$$E = -\frac{1}{2}\left[\sum_{kln} T_{kln} V_{kl} V_{ln} - \alpha\left(\sum_{kln(n\neq l)} V_{kl} V_{kn} + \sum_{klm(m\neq k)} V_{kl} V_{ml}\right) - \beta\left(\sum_{mn} V_{mn} - N_a\right)^2\right], \quad (1)$$

where V_{ij} is a pair of hits (*doublet*), i and j are the indices for the two hits. The first term in the square brackets reduces the energy when two doublets sharing a hit align on a straight line, and the length of the doublet is short. The second term gives a penalty when doublets share hits. The last term controls the number of doublets. In the high pileup condition, Eq. (1) causes a zigzag pattern, which looks like a flip of a sign of the curvature at each layer because of the missing penalty term to avoid such pattern[1]. Additionally, many fake candidates tend to be produced at a high pileup condition.

Following the previous study [13], we use *triplets*, which consists of three hits, as a qubit instead of a doublet, and a target function to be minimized is defined as follows,

$$E = \left(\sum_{i}^{N} \alpha_i T_i\right) - \left(\sum_{i,j} S_{ij} T_i T_j\right) + \left(\sum_{i,j} \zeta_{ij} T_i T_j\right), \quad (2)$$

[1] A method with additional penalty terms is proposed in Ref. [6].

where T_i represents whether a triplet is legitimate or not, and it is assigned as a qubit. This form is called a Quadratic Unconstrained Binary Optimization (QUBO) and used in the annealing framework. The first term constrains the number of triplets unable simultaneously and gives a penalty if the triplet seems to be a combinatorial fake, which is identified using impact parameters of a triplet,

$$\alpha_i = \frac{1}{2}\left(1 - e^{-d_0/C_{d_0}}\right) + \frac{1}{5}\left(1 - e^{-z_0/C_{z_0}}\right), \tag{3}$$

where C_{d_0} and C_{z_0} are normalization constants controlling the magnitude of the penalty. We assign 1.0 (0.5) mm as C_{d_0} (C_{z_0}) in this study. The values were determined to maximize the tracking performance (which is defined in Sect. 3.2).

The second term gives a reward if the relation of two triplets is consistent with a hypothesis that they belong to a same trajectory, and the hypothesis is evaluated using the curvature, direction, and the presence of a missing hit (*hole*) in the triplet. The coefficient (S_{ij}) is defined as

$$S_{ij} = \frac{1 - \frac{1}{2}(P^R_{i,j} + P^\theta_{i,j})}{(1 + H_i + H_j)^2}, \tag{4}$$

$$P^R_{i,j} = \frac{\left|(1/R)_i - (1/R)_j\right|}{C^R}, \tag{5}$$

$$P^\theta_{i,j} = \frac{max(\delta\theta_i,\ \delta\theta_j)}{C^\theta}, \tag{6}$$

where H_i is the number of holes in the i-th triplet, $(1/R)_i$ is the curvature of the i-th triplet, and $\delta\theta_i$ is the difference of polar angles between two doublets in the i-th triplet. The C^R and C^θ are the normalization constants controlling how much inconsistency is allowed. We assign 0.1 mm^{-1} (0.1 radians) as C^R (C^θ) in this study.

The third term gives a penalty to a triplet pair with shared hit. Since it is rare for a hit to be shared by two charged particles, a track having a shared hit tends to be a combinatorial fake. We use $\zeta_{ij} = 1$ for a pair of triplets which share the same hits, and $\zeta_{ij} = 0$ otherwise.

Eq. (2) can be written simply as follows

$$O(a, b, T) = \sum_i^N a_i T_i + \sum_i^N \sum_{j<i}^N b_{ij} T_i T_j \tag{7}$$

$$b_{ij} = \begin{cases} -S_{ij} & \text{if a pair of triplets align in sequence} \\ \zeta_{ij} & \text{if a pair of triplets share a hit} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

2.4 Triplet selection

A triplet selection is applied before a QUBO building to reduce the size of QUBO for efficient QUBO solving. It is required for a triplet that the number of holes (H_i) is less than two, the curvature($1/R$) is less than 0.8 [mm^{-1}], and the consistency of the polar angle within a triplet ($\delta\theta_i$) is less than 0.1. Additionally, it is required that there is a pair of triplets that align in sequence with sharing two hits and are consistent with originating from a same charged track with $P^R_{i,j} \leq 1$ and $S_{ij} > 0.2$.

3 D-Wave quantum annealer

3.1 QUBO solver

We use two software tools to solve a QUBO problem. The first one is qbsolv [16], which is a solver provided by the D-Wave for the D-Wave quantum annealing system [17]. The qbsolv splits a large QUBO problem into small sub-QUBOs to fit the D-Wave system. The solver finds a global minimum of the QUBO by iterating the process of (1) splitting into sub-QUBOs and finding globally the lowest energy state for each sub-QUBO using the D-Wave quantum annealer and (2) combining the sub-QUBO results and optimizing the solution locally using a classical tabu solver. In this study, we use D-Wave 2000Q system, which has 2048 qubits and the qubits connect with the other 5-6 qubits. The second software tool is neal [18], which is a classical simulated annealing solver. This solver is used for comparison of the results with the D-Wave system.

3.2 Performance

Figure 2 shows a set of doublets belonging to triplets with different colors representing different doublet types, and the doublets are projected onto a *x*-*y* plane perpendicular to the beam line. Before the triplet selection, there are 390,000 doublets, where more than 99 percent of doublets are fake. After the triplet selection, the number of doublets decreases to 2,445, but there remain fake doublets as shown in Figure 2 (a). After QUBO solving, almost all fake doublets disappear as shown in Figure 2 (b), where the number of reconstructed doublets is 1,424 with the purity of 98.5 percent and the efficiency of 96.4 percent. Here, the efficiency is defined as a ratio of the number of reconstructed and truth-matched doublets to the number of true doublets, and the purity is defined as a ratio of the number of reconstructed and truth-matched doublets to the number of reconstructed doublets.

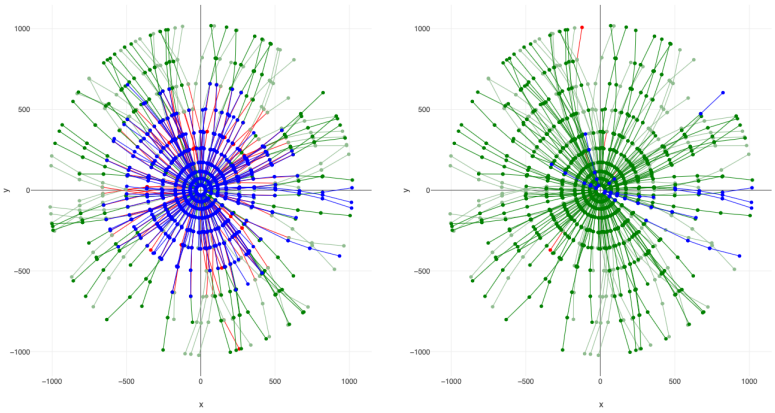
Figure 3(a) shows purity and efficiency as a function of the number of charged particles obtained by using the two solvers (qbsolv and neal). The lowest bin (number of particles = 818) corresponds to the 10 percent of the HL-LHC pileup condition, and the highest bin (= 6549) corresponds to the 80 percent of that. Our method works with more than 90 percent efficiency and more than 85 percent purity even at 80 percent of the HL-LHC pileup condition. Both solvers have similar performance at a low pileup environment, while the performance of the qbsolv with the D-Wave annealing machine is worse than the classical simulated annealing at a high pileup environment. Figure 3(b) shows the compositions of the doublets found after solving the QUBO. The missing doublets and fake doublets both increase at a high pileup environment.

4 Fujitsu Digital Annealer

The Fujitsu Digital Annealer [19] is an annealing device solving QUBO problems by classical simulated annealing using logic circuits on a chip. Compared to the D-Wave quantum annealer, it is easy to increase the QUBO size and the Digital Annealer works at a normal temperature.

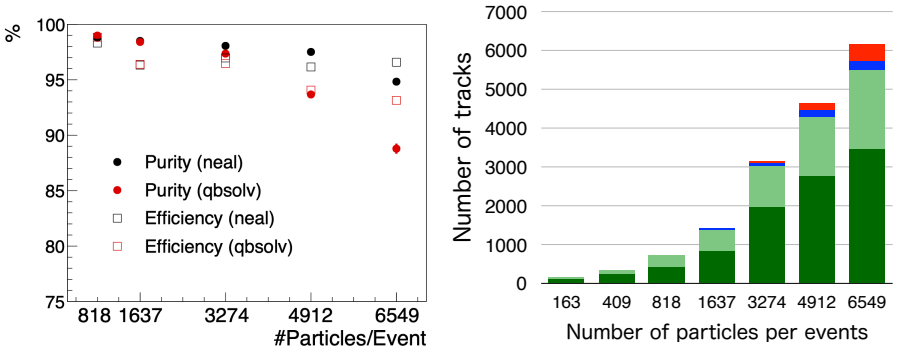
The use of classical simulated annealer for the tracking problem is demonstrated with using the 1st generation chip of the Fujitsu Digital Annealer [19], which has 1024 bits with full connection with other bits [2] and has 16-bit accuracy for a coupling weight in a QUBO. To fit the QUBO size to the limitation of the device, a QUBO is built by taking a narrow η

[2] As described in Sect. 3.1, qubits in D-Wave 2000Q system connects with the other 5-6 qubits (Chimera graph). Therefore, the Fujitsu Digital Annealer can embed much larger QUBO than D-Wave 2000Q.



(a) (b)

Figure 2: An example of doublets on x-y place in one collision event (a) before QUBO solving (b) after QUBO solving. Dark (light) green line shows a reconstructed doublet within (without) the momentum acceptance. Blue line shows a missing doublet, which should be reconstructed but not remain after QUBO solving. Red line shows a fake doublet, which is not associated with injected tracks and remains after QUBO solving. Since the categorization of doublets is evaluated after connecting the neighbor doublets, a badly connected doublet with wrong neighbor doublets is identified as a missing.

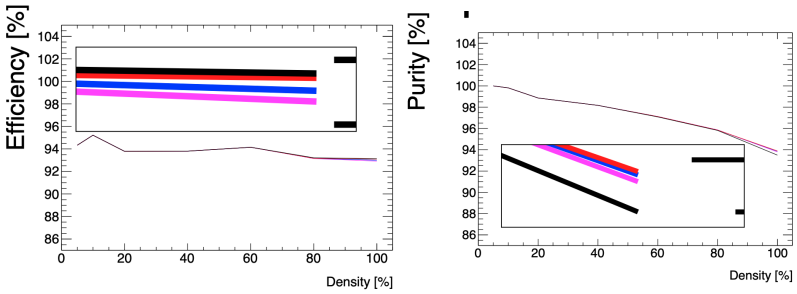


(a) (b)

Figure 3: (a) Purity (circle) and efficiency (square) as a function of the number of particles. Black (red) shows the results by neal (qbsolv) solver. (b) The number of tracks of each categories. Tracks are reconstructed from connected doublets. If a track contains less than five doublets, the track is rejected. A coloring is defined in a similar way in Figure 2.

region of the detector (width of 0.07) and shifting the η region iteratively with small overlaps of the width of 0.01.

Figure 4 shows the efficiency and purity of an event randomly selected for the performance evaluation as a function of the density. The performance of the Digital Annealer is nearly the same as the neal solver for all hit density conditions.



(a) (b)

Figure 4: Efficiency (a) and purity (b) as a function of a fraction of a hit density for a HL-LHC pileup environment. Black line is a result by the neal solver. Magenta, blue and red line show results by the Digital Annealer with different solver-configurations.

Table 1 shows the CPU time of the annealing by the neal solver, CPU time of a preprocessing/postprocessing by the Digital Annealer, and annealing time on the Digital Annealer Unit. A queue and network time are not included here. Also, a common preprocessing/postprocessing time (triplet selection/QUBO building/track formation) are not shown in the table. The annealing time on the Digital Annealer is independent of the hit density, while the CPU time depends on the hit density. The dominant part of the computing time for the Digital Annealer is CPU time in the case of the full density.

Table 1: A comparison of the compute time of the Digital Annealer and the neal solver. N_{slice} is the number of η slices. A queue and network time of the Digital Annealer are not included in this table.

Density [%]	N_{slice}	DA [sec]		neal [sec]
		CPU time	Anneal time	total time
5	46	0.09	0.29	0.27
10	68	0.15	0.42	0.66
20	71	0.22	0.44	1.29
40	74	0.52	0.45	2.46
60	73	0.94	0.45	4.29
80	74	1.79	0.46	7.49
100	74	3.73	0.45	12.87

5 Conclusion

We demonstrated a new method of track finding with annealing device. A QUBO is built based on triplets, considering the relation between triplets and the property of the triplet

itself. The track finding is performed by searching for the lowest energy state under the Hamiltonian using a D-Wave quantum solver, a classical simulated annealing solver running on CPU, and the Fujitsu Digital Annealer. In a high pileup condition up to 80 percent of the HL-LHC pileup condition, the efficiency is more than 85 percent, and the purity is more than 90 percent for the qbsolv, while the performance of the neal solver is better than the qbsolv. The performance of the Fujitsu Digital Annealer is evaluated with splitting data into η slices and is nearly the same as the neal solver.

The annealing devices are one of the possible approaches for the track pattern recognition task. However, to utilize D-Wave quantum device with its quantum advantages or to reduce the total time of the track finding, there are some considerable issues as discussed in Ref. [13]. Further studies are necessary to use the annealing techniques in the practical track finding tasks.

References

[1] G. Apollinari, I. Béjar Alonso, O. Brüning, P. Fessia, M. Lamont, L. Rossi, L. Tavian, *High-Luminosity Large Hadron Collider (HL-LHC): Technical Design Report V. 0.1*, CERN Yellow Reports: Monographs (CERN, Geneva, 2017), https://cds.cern.ch/record/2284929

[2] *TrackML Particle Tracking Challenge*, https://www.kaggle.com/c/trackml-particle-identification

[3] S. Amrouche, L. Basara, P. Calafiura, V. Estrade, S. Farrell, D.R. Ferreira, L. Finnie, N. Finnie, C. Germain, V.V. Gligorov et al., *The Tracking Machine Learning Challenge: Accuracy Phase*, in *The NeurIPS '18 Competition*, edited by S. Escalera, R. Herbrich (Springer International Publishing, Cham, 2020), pp. 231–264, ISBN 978-3-030-29135-8

[4] *HEP advanced tracking algorithms with cross-cutting applications (Project HEP.TrkX)*, https://heptrkx.github.io/

[5] Shapoval, Illya, Calafiura, Paolo, EPJ Web Conf. **214**, 01012 (2019)

[6] A. Zlokapa, A. Anand, J.R. Vlimant, J.M. Duarte, J. Job, D. Lidar, M. Spiropulu (2019), 1908.04475

[7] S. Das, A.J. Wildridge, S.B. Vaidya, A. Jung (2019), 1903.08879

[8] C. Tüysüz, F. Carminati, B. Demirköz, D. Dobos, F. Fracas, K. Novotny, K. Potamianos, S. Vallecorsa, J.R. Vlimant, *A Quantum Graph Neural Network Approach to Particle Track Reconstruction* (2020), 2007.06868

[9] A.W. Harrow, A. Hassidim, S. Lloyd, Phys. Rev. Lett. **103**, 150502 (2009)

[10] P.W. Shor, *Algorithms for quantum computation: discrete logarithms and factoring*, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (1994), pp. 124–134

[11] F. Arute, K. Arya, R. Babbush, D. Bacon, J.C. Bardin, R. Barends, R. Biswas, S. Boixo, F.G.S.L. Brandao, D.A. Buell et al., Nature **574**, 505 (2019)

[12] *IBM Research Blog*, https://www.ibm.com/blogs/research/2019/09/quantum-computation-center/

[13] F. Bapst, W. Bhimji, P. Calafiura, H. Gray, W. Lavrijsen, L. Linder, A. Smith, Computing and Software for Big Science **4**, 1 (2019)

[14] *ATLAS Computing Public Result*, https://twiki.cern.ch/twiki/bin/view/AtlasPublic/ComputingandSoftwarePublicResults

[15] G. Stimpfl-Abele, L. Garrido, Comput. Phys. Commun. **64**, 46 (1991)

[16] *Dwave qbsolv*, https://github.com/dwavesystems/qbsolv

[17] *DWave System*, https://www.dwavesys.com/

[18] *Dwave simulated annealing sampler*, https://github.com/dwavesystems/dwave-neal

[19] *Fujitsu Digital Annealer*, https://www.fujitsu.com/global/services/business-services/digital-annealer/index.html