A Pattern Recognition Algorithm for Quantum Annealers

Frédéric Bapst[1] [ID] · **Wahid Bhimji**[2] · **Paolo Calafiura**[2] · **Heather Gray**[2,3] [ID] · **Wim Lavrijsen**[2] · **Lucy Linder**[1] · **Alex Smith**[3]

Abstract

The reconstruction of charged particles will be a key computing challenge for the high-luminosity Large Hadron Collider (HL-LHC) where increased data rates lead to a large increase in running time for current pattern recognition algorithms. An alternative approach explored here expresses pattern recognition as a quadratic unconstrained binary optimization (QUBO), which allows algorithms to be run on classical and quantum annealers. While the overall timing of the proposed approach and its scaling has still to be measured and studied, we demonstrate that, in terms of efficiency and purity, the same physics performance of the LHC tracking algorithms can be achieved. More research will be needed to achieve comparable performance in HL-LHC conditions, as increasing track density decreases the purity of the QUBO track segment classifier.

Keywords Quantum annealing · Pattern recognition · HEP particle tracking

Introduction

Early quantum computers are rapidly being made available both in the cloud and as prototypes in academic and industrial settings. These devices span the range from D-Wave [1] commercial quantum annealers to gate-based quantum processor prototypes based on a wide range of promising technologies [2]. Quantum computing holds the potential for super-polynomial speedups and large decrease in energy usage, if suitable algorithms can be developed. It is therefore crucial to start identifying algorithms and applications for high-energy physics, to be ready for when quantum computing becomes mainstream and to provide input about what features are needed in quantum computers to solve problems in high-energy physics.

The reconstruction of charged particles will be a key computing challenge for the high-luminosity Large Hadron Collider (HL-LHC) where increased data rates lead to a large

increase in running time for conventional pattern recognition algorithms. Conventional algorithms [3, 4], which are based on combinatorial track seeding and building, scale quadratically or worse as a function of the detector occupancy.

We present an alternative approach, one that expresses pattern recognition as a quadratic unconstrained binary optimization (QUBO; a NP-hard problem) using annealing, a process to find the global minimum of an objective function—in our case a quadratic function over binary variables based on the algorithm introduced in Ref. [5] following ideas in Refs. [6, 7]. The term annealing is inspired by the metallurgic process of repeated heating and cooling to remove dislocations in the lattice structure. Likewise as used here, the annealing optimization process uses random "thermal" fluctuations to find better results of the objective function, combined with a "cooling" which progressively reduces the probability of accepting a worse result. Quantum annealing is grounded in the adiabatic theorem: a system will remain in its eigenstate if perturbations that act on it are slow, and small enough not to span the gap between the ground and first excited states [8]. Thus, it is possible to initialize a quantum annealer with a simple ground state Hamiltonian and evolve it adiabatically to the desired, complex, problem Hamiltonian. After evolution, quantum fluctuations, such as tunneling, bring the annealer into the ground state of the latter, representing the global minimum solution of the problem [9]. All steps of quantum annealing operate on the system as a whole and the total time required is typically

✉ Lucy Linder
lucy.linder@hefr.ch

1   School of Engineering and Architecture of Fribourg, Switzerland, iCoSys Institute, HES-SO Fribourg, Fribourg, Switzerland

2   Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

3   Department of Physics, University of California, Berkeley, CA 94720, USA

bounded for a given device. Thus, as long as the problem fits on the annealer, the total running time should be constant, and it is hoped that a large enough quantum system, running an intricate problem, can outperform a software-based one.

We test our approach using annealing both in software simulation and by running on a D-Wave quantum computer. We use a dataset representative of the expected conditions at the HL-LHC from the TrackML challenge [10]. We study the performance of the algorithm as a function of the particle multiplicity. We do not expect to obtain speed improvements because the size of the currently available annealers is smaller than the scale of our problem.

Methodology

Pattern Recognition: General Considerations

The goal of pattern recognition is to identify groups of detector hits to form tracks. Track trajectories are parameterized using the following five parameters: d_0, z_0, ϕ_0, $\cot\theta$, and q/p_T.[1] The transverse impact parameter, d_0, is the distance of closest approach of the helix to the chosen reference point (e.g., the primary vertex) in the x-y plane. The longitudinal impact parameter, z_0, is the z coordinate of the track at the point of closest approach. The azimuthal angle, ϕ_0, is the angle of the track in the x-y at the point of closest approach. The polar angle, $\cot\theta$ is the inverse slope of the track in the r-z plane. The curvature, q/p_T, is the inverse of the transverse momentum with the sign determined by the charge of the particle.

Neglecting noise and multiple scattering, most particle tracks of physics interest, particularly those with high p_T, exhibit the following properties:

- The hits follow an arc of a helix in the x-y plane with a large radius of curvature or small q/p_T;
- The hits follow a straight line in the r-z plane;
- Most hits lie on consecutive layers: there are few to no missing hits (holes).

Track candidates with fewer than five hits are predominantly fake tracks, which do not correspond to a true particle trajectory. While tracks can share hits, we impose the constraint

from Ref. [10] that any one hit can belong to at most one track.

Algorithm Goals

The algorithm presented in this paper encodes a classification problem. Following Ref. [5], tracks are constructed from n consecutive hits, leading to $n-1$ doublets. Given the large set of potential doublets from hits in the detector, the goal of the algorithm is to determine which subset belongs to the trajectories of charged particles. The algorithm aims to preserve the efficiency, but improve the purity of the input doublet set.

Triplets and Quadruplets

We follow a similar approach to Ref. [5], but use triplets instead of doublets. In addition to improving the performance at high multiplicity, this allows us to calculate and use track properties.

A triplet, denoted T^{abc}, is a set of three hits (a, b, c) or a pair of consecutive doublets (a, b and b, c), ordered by increasing transverse radius (R). Two triplets T^{abc} (of hits a, b, c) and T^{def} (of hits d, e, f), can be combined to form a quadruplet if $b = d \wedge c = e$ or a quintet if $c = d$. If they share any other hit, the triplets are marked as being in conflict. A set of n consecutive hits will result in $n-2$ triplets and $n-3$ quadruplets.

Key triplet T_i^{abc} properties are the number of holes H_i; the curvature, q/p_T; and $\delta\theta$ the difference in polar angle between the doublets.

The strength S quantifies the compatibility of the track parameters between the two triplets in a quadruplet (T_i, T_j):

$$S(T_i, T_j) = z_1 \frac{z_2 \left(1 - |\delta(q/p_{Ti}, q/p_{Tj})|\right)^{z_3}}{(1 + H_i + H_j)^{z_5}} \tag{1}$$

$$+ \frac{(1 - z_2)\left(1 - \max(\delta\theta_i, \delta\theta_j)\right)^{z_4}}{(1 + H_i + H_j)^{z_5}} \tag{2}$$

where z_2 encodes the relative importance of the curvature with respect to $\delta\theta$. The other parameters (z_1, z_3, z_4, z_5) are unbounded constants that require problem-specific tuning. The parameters are set to favor high p_T tracks. In its simplest form, we have $z_2 = 0.5$ (equal weights), $z_5 = 2$, and all other constants set to 1:

$$S(T_i, T_j) = \frac{1 - \frac{1}{2}(|\delta(q/p_{Ti}, q/p_{Tj})| + \max(\delta\theta_i, \delta\theta_j))}{(1 + H_i + H_j)^2} \tag{3}$$
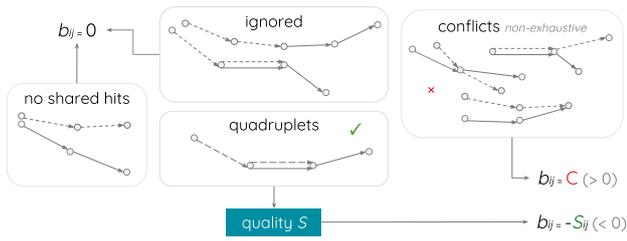
---

[1] We use a right-handed coordinate system with its origin at the nominal interaction point (IP) in the center of the detector. The x-axis points from the IP to the center of the LHC ring, the y-axis points upward, and the z-axis coincides with the axis of the beam pipe. Cylindrical coordinates (r, ϕ) are used in the transverse plane, ϕ being the azimuthal angle around the beam pipe. The polar angle θ lies in the r-z plane.

Fig. 1 The value assigned to the QUBO quadratic weights b_{ij} for different configurations of the pairs of triplets T_i and T_j. See text for details

Definition of the Quadratic Unconstrained Binary Optimization

The QUBO is configured to identify the best pairs of triplets. It has two components: a linear term that weighs the quality of individual triplets and a quadratic term used to express relationships between pairs of triplets. In our case, the objective function to minimize becomes:

$$O(a, b, T) = \sum_{i=1}^{N} a_i T_i + \sum_{i}^{N} \sum_{j<i}^{N} b_{ij} T_i T_j \quad T_i, T_j \in \{0, 1\} \quad (4)$$

where T are all potential triplets, a_i are the bias weights, and b_{ij} the coupling strengths computed from the relation between the triplets T_i and T_j. The bias weights and the coupling strengths define the Hamiltonian. Minimizing the QUBO is equivalent to finding the ground state of the Hamiltonian.

All bias weights are set to be identical $a_i = \alpha$, which means all triplets have equal a priori probability to belong to a particle track. Our objective function therefore depends solely[2] on the triplet–triplet coupling strength b_{ij}. If the triplets form a valid quadruplet, the coupling strength is negative and equal to the quadruplet quality $S(T_i, T_j)$ (Eq. 3). If the two triplets are in conflict, the coupling is a positive constant $b_{ij} = \zeta$ that disfavors a solution with $T_i = T_j = 1$. Finally, if the triplets have no relationship (meaning, no shared hits), the coupling is set to zero. This is illustrated in Fig. 1 and represented in Eq. 5.

$$b_{ij} = \begin{cases} -S(Ti, Tj), & \text{if } (T_i, T_j) \text{ form a quadruplet}, \\ \zeta & \text{if } (T_i, T_j) \text{ are in conflict}, \\ 0 & \text{otherwise}. \end{cases} \quad (5)$$

As is clear from Eq. 5, the choice of constants in Eq. 1 determines the functional behavior of b_{ij}. The larger the conflict strength ζ the lower the number of conflicts, but too large values risk discontinuities in the energy landscape, increasing time to convergence. Furthermore, the D-Wave machines limit the value of b_{ij}, and thus ζ, to between -2 and 2 (with a restricted precision, so scaling is not a fix either).

Dataset Selection

By design, the algorithm does not favor any particular momentum range. However, to limit the size of the QUBO, we focus on high p_T tracks ($p_T \geq 1$ GeV), which are the most relevant for physics analysis at the HL-LHC.

A triplet T_i is created if and only if:

$$H_i \leq 1$$
$$|(q/p_T)_i| \leq 8 \times 10^{-4} \text{ GeV}^{-1},$$
$$\delta\theta_i \leq 0.1 \text{ rad}$$

And a quadruplet (T_i, T_j) is created if and only if:

$$|\delta((q/p_T)_i, (q/p_T)_j)| \leq 1 \times 10^{-4} \text{ GeV}^{-1},$$
$$S(T_i, T_j) > 0.2$$

Triplets that are not part of any quadruplet or whose longest potential track has less than five hits are not considered.

Experimental Setup

Dataset

The TrackML dataset is representative of future high-energy physics experiments at the HL-LHC. It anticipates the HL-LHC multiplicities planned for after 2026. Both the low p_T cut (150 MeV) and high luminosity (200 μ) make pattern recognition within this dataset a challenging task. We simplify the dataset by focusing on the barrel (experiment mid-section, with detectors mostly parallel to the beamline) region of the detector, i.e., hits in the end caps (both experiment end sections, with detectors mostly transverse to the beamline) are removed. If a particle makes multiple energy deposits in a single layer, all but one energy deposits are removed. Hits from particles with $p_T < 1$ GeV and particles with less than five hits are kept and thus part of the pattern recognition, but are not taken into account when computing the performance metrics. Events are split by randomly selecting a fraction of particles and an equal fraction of noise to generate datasets with different detector occupancies yet similar characteristics. We note that this is not fully equivalent to a lower multiplicity event because such a procedure selects a fraction of the tracks in a pile-up event rather than a fraction of the pile-up events.

---

[2] No difference was observed when shifting the bias weight α by a small amount.

Metrics

The performance is assessed using purity and efficiency,[3] which are computed on the final set of doublets. This provides a good estimate of the quality of the model as a doublet classifier, but does not account for the difference in importance between track candidates for physics. The TrackML score [10] is used as a complementary metric as it includes weights to favour tracks with higher p_T, which play a larger role in physics performance.

The efficiency and purity are defined as follows:

$$\text{Efficiency} = \frac{D^{\text{rec}}_{\text{matched}}}{D^{\text{true}}} \tag{6}$$

$$\text{Purity} = \frac{D^{\text{rec}}_{\text{matched}}}{D^{\text{rec}} - D^{\text{rec}}_{\text{oa}}} \tag{7}$$

The number of true doublets (D^{true}) only includes those with $p_T > 1$ GeV, which deposit at least five hits in the detector barrel. Reconstructed doublets (D^{rec}) are matched to true doublets using truth information ($D^{\text{rec}}_{\text{matched}}$). Reconstructed doublets matched to true doublets, but with either $p_T \leq 1$ GeV or less than five hits in the detector barrel ($D^{\text{rec}}_{\text{oa}}$) are excluded from the purity.

Initial Doublets

The initial set of doublets is generated using an adaptation as a Python library of the ATLAS online track seeding code [11]. It was tuned to ensure an efficiency above 99% for high p_T tracks, but has a purity below 0.5%.

QUBO Solver

qbsolv [12] is a tool developed by D-Wave to solve larger and more densely connected QUBOs than currently supported by the D-Wave hardware. It uses an iterative hybrid classical/quantum approach with multiple trials. In each trial, the QUBO is split into smaller instances that are submitted to a sub-QUBO solver for global optimization. Results are combined and a tabu search [13] is performed for local optimization. The sub-QUBO solver is either a D-Wave system or a software-based solver. Using this setup, running qbsolv on a classical system has the same workflow as running qbsolv with D-Wave, making it an effective simulator. D-Wave also provides NEAL [14], a standalone software-only annealer, which we use for comparison studies.
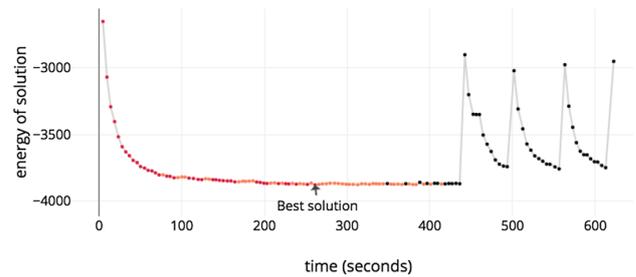

Fig. 2 An example event showing the convergence towards the solution in each qbsolv loop. The algorithm occasionally jumps away from the current best solution which is an intended feature to avoid local minima
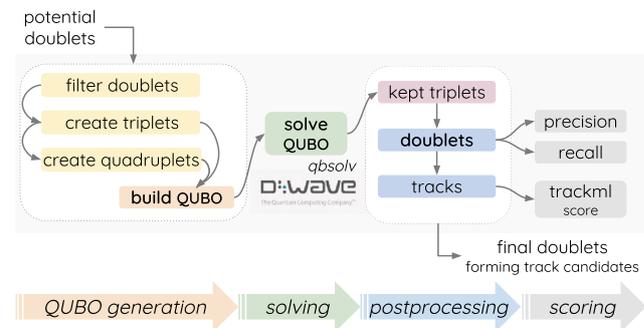

Fig. 3 Overview of the steps in the algorithm

The number of sub-QUBOs that are created can be controlled by restricting the size of the number of logical qubits that can be used per sub-QUBO. We use the default value of 47 for both the simulator and the D-Wave, as it worked well: larger or smaller numbers can result in a failed mapping, and a subsequent abort of the run. Another qbsolv parameter we have tuned is the number of times the main loop of the algorithm is repeated before stopping. The default value of 50 was too conservative for our problem which converges smoothly to the optimal solution (Fig. 2). Reducing that value to 10 sped up the solving step without any performance loss. Other qbsolv command line parameters do not appear to influence the algorithm performance, and were also left at their default values.

We ran our simulations on the Cori [15] supercomputer at NERSC, experiments on the Ising D-Wave 2X machine at Los Alamos National Laboratory (with 1000 qubits), and tests on the D-Wave LEAP cloud service. The number of iterations and D-Wave samplings was limited to 10.

Complete Algorithm

Figure 3 illustrates the steps in the algorithm [16]. The initial doublets are combined into triplets and quadruplets,
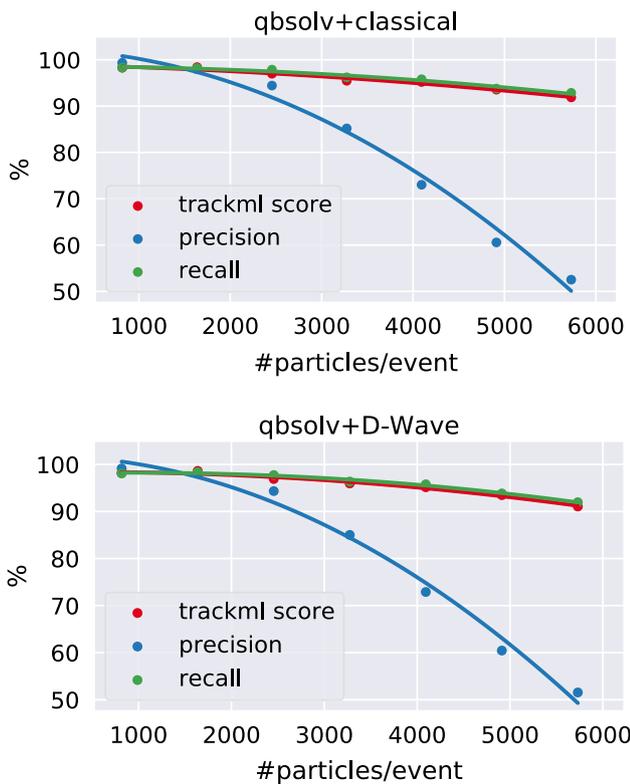
Fig. 4 The performance of classical simulator (top) and D-Wave (bottom), as measured by TrackML score (red), purity (blue), and efficiency (green), as a function of particle multiplicity
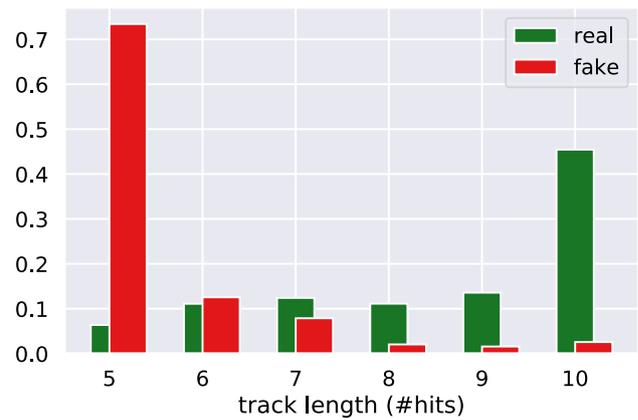

Fig. 5 The fraction of real (green) and fake (red) doublets as a function of the number of hits on track

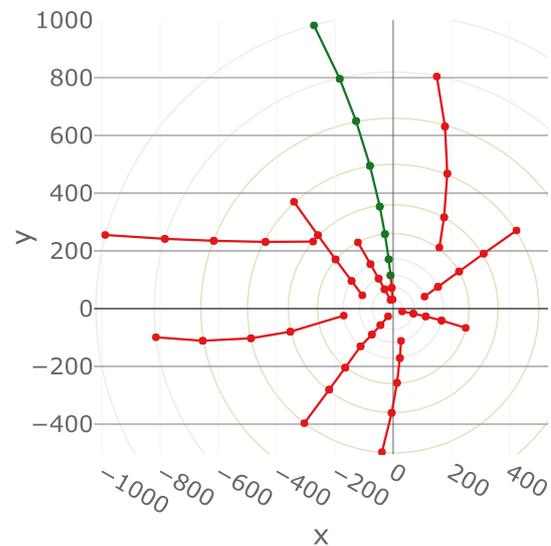**Fig. 6** Selected real (green) and fake (red) track candidates from a low multiplicity event

after satisfying the requirements from Sect. 2.4. Details of the QUBO building including the quality cuts applied to triplets and quadruplets are discussed in [16]. The QUBO is generated and sampled using qbsolv. The post-processing phase includes converting the triplets into doublets, removing duplicates and removing any doublets with unresolved conflicts. The track candidates are reconstructed from the doublets, and track candidates with less than five hits are discarded. Finally, performance metrics are computed and the set of final doublets corresponding to the track candidates is returned.

Results

We chose three events from the dataset containing 10K, 12K and 14K particles plus noise, with the latter being the highest multiplicity event in the dataset. We sample from these events to construct sets ranging from O(1K) to O(7K) particles. Each set is constructed by taking a fixed fraction of the particles and the noise in that event.

Algorithmic Performance

We use purity and efficiency, as defined in Sect. 3.2, to assess the algorithmic performance. Figure 4 shows these metrics as a function of the particle multiplicity. Efficiency and the TrackML score are well above 90% across the range, with the purity starting close to 100%, but dropping to about 50% for the highest occupancies considered. As the purity drops with increasing occupancy, the number of fake doublets rises. The D-Wave machine results are well reproduced by the simulation. The reproducibility of the results was checked by repeating the qsolving step on D-Wave for the same QUBO.

Figure 5 shows the fraction of real and fake doublets as a function of the number of hits on tracks. As the fake tracks
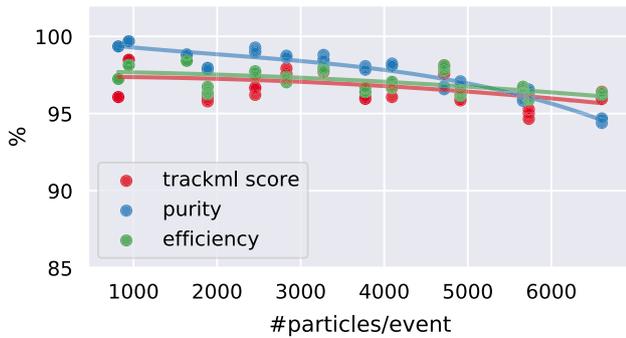
Fig. 7 The performance of the classical simulator as measured by TrackML score (red), purity (blue), and efficiency (green), as a function of particle multiplicity when including a bias term based on the impact parameter of the triplet

(i.e., tracks containing one or more fake doublets, see Fig. 6) tend to have fewer hits, the purity can be improved, with minimal efficiency loss, by requiring barrel tracks to have at least six hits.

The purity can be improved to above 90% (see Fig. 7) by adding new properties to the QUBO such as the extrapolated track perigee or impact parameters, but at the cost of biasing the algorithm against tracks with large impact parameters.

We find that the results from the simulator match those of the D-Wave machine rather well. This allows us to use the simulation to tune the parameters for the experiments on the D-Wave machine. No significant impact of noise from the machine on the final results is observed.

Throughput and Timing

Our current experimental setup does not allow to perform detailed timing studies. This is because the devices used are shared, accessed remotely and inherently stochastic.

Today, D-Wave devices are not fully connected and thus require synthesis of logical qubits, via a process named "minor embedding" [17]. In our setting, the process that takes the initial set of doublets and generates the QUBO placement onto the D-Wave is approximately linear over the range of input doublets considered. It takes up to an hour on the largest dataset, which we view as a limitation of the current approach. However, we expect that the run time would be improved by code optimization including parallelization and by exploiting more advanced track seeding algorithms. All QUBO solvers scale similarly, with a superlinear running time as a function of occupancy. NEAL is two orders of magnitude faster than qbsolv.

On D-Wave, the annealing is run ten times for each sub-QUBO to reduce the impact of noise. There is significant initial setup time on D-Wave, as well as additional overhead due to the time required for minor embedding.

In the configuration described in Sect. 3.4, the quantum annealer converges reliably to a solution, likely due to the smoothness of the energy function $O(a, b, T)$ (Eq. 4). Varying the constant bias α between $[-0.1, 0.1]$, and the conflict term ζ between $[0.8, 2.0]$ has little impact on the algorithm purity and efficiency, that change by less than 3% across the intensity (#particles/event) range.

Related Work

Ref. [18] shows that quantum Hopfield associative memory can be implemented and trained on a D-Wave computer. When training a Hopfield network, the optimization goal is to find the set of connection weights that minimizes the network energy for a given set of training patterns. In this work, we used charged particle properties to determine a set of weights and then the set of patterns that minimize the QUBO energy.

Ventura's quantum associative memory (QuAM) is a quantum pattern matching algorithm derived from Grover's search [19] providing exponential storage capacity [20]. That algorithm targets pattern recognition algorithms in trigger detectors, while the algorithm discussed here targets offline pattern recognition.

Discussion

The main algorithmic innovation reported here is the introduction of a triplet-based QUBO. The richer feature set of a triplet allows the QUBO to achieve greater than 90% efficiency at track densities which are comparable to HL-LHC[4]. The binary constraints used in the QUBO are based on matching the p_T and θ track parameters between two triplets. Improvement can be achieved using the full track covariance matrix. Further improvements may come from more refined hyperparameter tuning, integration of the detailed geometry and magnetic field description, tuning the preselection according to the detector location and topologies of the triplets and the use of quadruplets instead of triplets in the QUBO.

When considering throughput, the timing is driven by partitioning the QUBO to fit on the available hardware, given the limited connectivity and the available number of qubits. The running time of individual sub-QUBOs was observed to be constant. The overall execution time was found to scale with the number of sub-QUBOs. Because of this, we do not currently observe an advantage in running on the D-Wave system. We observed that our large QUBO

---

[4] And two orders of magnitude higher than in Ref. [5]

instances are processed quite efficiently with a particular classical solver. In addition, formulating the problem as a QUBO has the additional advantage of also being compatible with other kinds of special hardware dedicated to the Ising model.

Conclusion

We ran pattern recognition on events representative of expected conditions at the HL-LHC on a D-Wave quantum computer using qbsolv, and provided a detailed analysis of the physics performance of the algorithm. At low track multiplicity, we obtain results with purity and efficiency comparable to current algorithms. We were able to run on events with as many as 6600 tracks. A very good performance was obtained with up to approximately 2000 particles per event, after which efficiency remains high, but purity starts to drop. Ideas for future algorithmic improvements were also explored. Further investigations would be required to study and optimize the timing performance of such algorithms.

Compliance with Ethical Standards

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. D-Wave systems. https://www.dwavesys.com. Accessed 7 Dec 2019
2. Preskill J (2018) Quantum computing in the NISQ era and beyond. arXiv e-print arXiv:1801.00862
3. Cornelissen T, Elsing M, Gavrilenko I, Liebig W, Moyse E, Salzburger A (2008) The new ATLAS track reconstruction (NEWT). J Phys Conf Ser 119(3):032014. https://doi.org/10.1088/1742-6596/119/3/032014
4. Collaboration CMS (2014) Description and performance of track and primary-vertex reconstruction with the CMS tracker. JINST 9(10):P10009. https://doi.org/10.1088/1748-0221/9/10/P10009
5. Stimpfl-Abele G, Garrido L (1991) Fast track finding with neural networks. Comput Phys Commun 64:46. https://doi.org/10.1016/0010-4655(91)90048-P
6. Denby BH (1988) Neural networks and cellular automata in experimental high-energy physics. Comput Phys Commun 49:429. https://doi.org/10.1016/0010-4655(88)90004-5
7. Peterson C (1989) Track finding with neural networks. Nucl Instrum Meth A279:537. https://doi.org/10.1016/0168-9002(89)91300-4
8. Morita S, Nishimori H (2008) Mathematical foundation of quantum annealing. J Math Phys 49(12):125210. https://doi.org/10.1063/1.2995837
9. Das A, Chakrabarti BK, Stinchcombe RB (2005) Quantum annealing in a kinetically constrained system. Phys Rev E 72:026701. https://doi.org/10.1103/PhysRevE.72.026701
10. Kaggle (2018) Trackml particle challenge. https://www.kaggle.com/c/trackml-particle-identification. Accessed 7 Dec 2019
11. Delgado AT, Emeliyanov D (2016) Nuclear science symposium, medical imaging conference and room-temperature semiconductor detector workshop (NSS/MIC/RTSD). IEEE, pp 1–6
12. Booth M, Reinhardt S, Roy A (2017) Partitioning optimization problems for hybrid classical/quantum execution. D-Wave Technical Report. https://www.dwavesys.com/sites/default/files/partitioning_QUBOs_for_quantum_acceleration-2.pdf. Accessed 7 Dec 2019
13. Glover F (1986) Future paths for integer programming and links to artificial intelligence
14. D-Wave neal. https://docs.ocean.dwavesys.com/projects/neal. Accessed 7 Dec 2019
15. Cori. https://www.nersc.gov/systems/cori. Accessed 7 Dec 2019
16. Linder L (2019) Hepqpr.qallse repository. https://github.com/derlin/hepqpr-qallse. Accessed 7 Dec 2019
17. Headquarters C (2013) Programming with D-Wave: map coloring problem
18. Seddiqi H, Humble T (2014) Adiabatic quantum optimization for associative memory recall. Front Phys. https://doi.org/10.3389/fphy.2014.00079
19. Ventura D, Martinez T (2000) Quantum associative memory. Inf Sci 124(1):273
20. Shapoval I, Calafiura P (2019) 23rd International conference on computing in high energy and nuclear physics (CHEP 2018), Sofia, Bulgaria, July 9–13, 2018