Towards the world fastest blockchain with quantum annealing

Kota Mikami Shiba Sotaro
k.mitkami@simia.info shiba.sota@simia.info

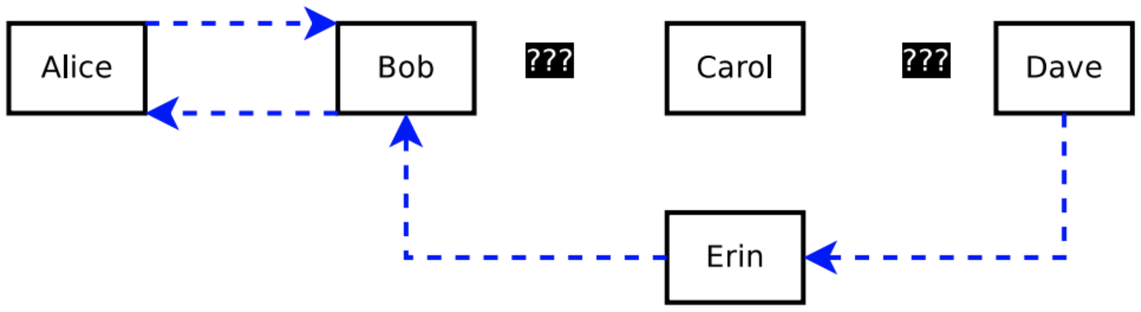
Simia Inc & University of Tokyo

Abstract—The use of quantum computing in graph community detection and regularity checking related to Szemeredi's Regularity Lemma (SRL) are demonstrated with D-Wave Systems' quantum annealer and simulations. We demonstrate the capability of quantum computing in solving hard problems relevant to big data. A new community detection algorithm based on SRL is also introduced and tested. In worst case scenario of regularity check we use Grover's algorithm and quantum phase estimation algorithm, in order to speed-up computations using a quantum gate computers. I

I. INTRODUCTION

We are entering the exciting era of quantum computing. There is hope that this new computing paradigm is also useful in studying hard problems in the analysis of large graphs emerging from big data. The use of quantum computing needs a new mindset. Probably the simplest avenue in this direction is the socalled quantum adiabatic computing and quantum annealing in particular which can be used in almost any optimization task. Quantum gate computing could be used for many more problems than a quantum annealer, but there each algorithm is an untrivial milestone in itself like the celebrated Shor's algorithm. The quantum annealing hardware is reaching over 5000 qubits (qubits are quantum objects that replace bits in ordinary computation) in the near future while gate computers are developing at the somehow more modest pace. The D-Wave Systems company has made quantum annealing available as a cloud service allowing experiments with over 2000 qubits as * corresponding author well as an easy to use interface to the system. Hybrid classicalquantum algorithms, available also for D-Wave machines, make solving larger problems possible. In this work, we consider use of quantum annealing for graph partitioning and, in particular, graph community detection with a new algorithm. We hope that our work will motivate other similar studies in the big data area. Our aim is to demonstrate the potential of quantum computing in analysing large graphs. Such an approach is likely to push the boundary of graph sizes in which good quality solutions can be found. We also demonstrate how quantum annealers are used in concrete cases.

A starting point of our work is Szemeredi's Regularity ´ Lemma (SRL), a cornerstone of extremal graph theory, see e.g. [4]. SRL justifies a kind of stochastic block model structure of bounded complexity for all large graphs. SRL has had a great impact in the theoretical study of large graphs and that is why it can have a decisive role in future big data analysis as well. SRL's key concept is an -regular bipartite graph. It is a bipartite graph in which link density deviations in any subgraphs are bounded by some positive .



This means that such a bipartite graph is close to random one. In SRL, the -parameter can be chosen to be arbitrarily small. SRL states, roughly speaking, that any large graph has a partitioning of nodes to a bounded number of sets in which links between parts follow the -regularity. Regular partitioning can be found in polynomial time. However, deciding -regularity of a bipartite graph is co-NPcomplete problem. We show that the regularity check is a arXiv:2006.16702v1 [cs.ET] 30 Jun 2020 binary quadratic optimization problem. It has the form that can be solved with the D-Wave quantum annealer. As a result, such optimization can be a very hard problem that can be of interest to test efficiency of quantum annealers and we use it as an example of a hard problem arising in large graph analysis. It appears that the same optimization task, as used in regularity check, can be used to find communities of an arbitrary graph. This novel algorithm does not need any parameters besides the adjacency matrix. The stochastic block model of communities can be seen as a particular realization of regular partition. In future we shall study a more general case of SRL from this point of view.

The suggested algorithm has some advantages over implementing the standard community detection algorithm on D-Wave [16]. Namely, it requires only 1 qubit per graph node and no prior knowledge of the number of communities. In standard approach, each node requires k times more qubits, in which k is the maximal number of communities. Since qubits are scarce resource, this difference is significant. We anticipate that quantum annealing can produce better quality solutions for large graph problems than classical computation. Interestingly, in [16] evidence pointing to this direction was already found. The quantum community detection algorithm found the best quality solution, measured in socalled modularity metrics, compared to any previous method. This was the case of well-known test graph with only 34 nodes, so-called Zachary Karate Club graph. Another point could be that such good solutions can be found in larger scales than is possible with classical computing. We test our ideas using D-Wave System and simulations. We also consider the performance of our community detection algorithm using stochastic block models and discuss further challenges.

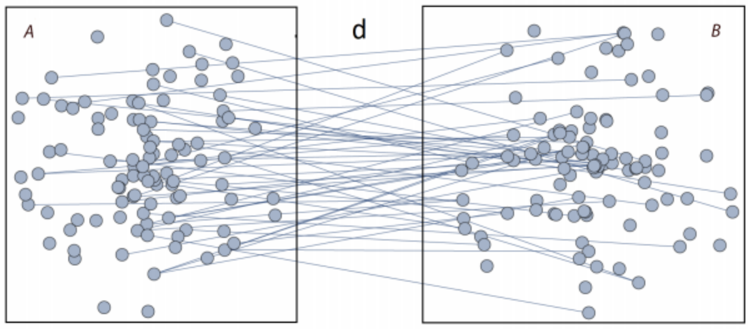
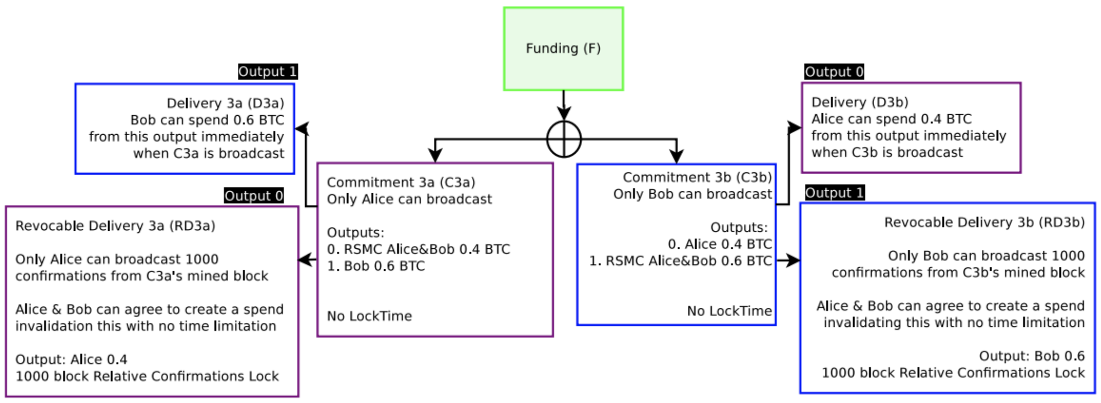


Fig. 1. A bipartite graph, where links can exist only between nodes in sets A and B; link density d is fraction of pairs $(i, j) \in A \times B$ that have links.

II. REGULARITY CHECK AS AN OPTIMIZATION PROBLEM

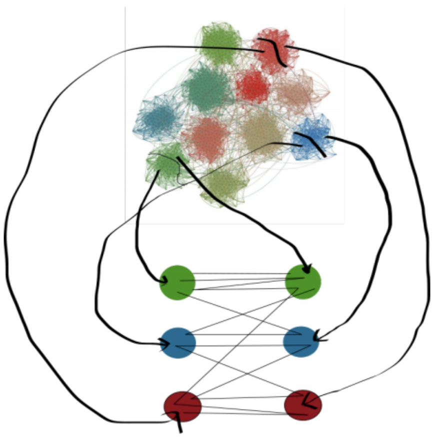
Let G = G(A, B, d(A, B)) denote a bipartite graph, in which the set of nodes of V , is divided into two disjoint sets A and B, see Fig. 1. The number of nodes (cardinality) of a set of nodes X, is denoted as |X|. The number of links connecting two arbitrary subsets of V , X and Y is denoted as e(X, Y). Similarly the link density between two disjoint node sets X and Y is by definition: d(X, Y) = e(X, Y) |X||Y | The binary adjacency matrix of G is denoted as A. Value (A)i,j is one only if there is a link between nodes i and j. A bipartite graph G(A, B, d(A, B)) is called -regular if, for all subsets X ⊂ A, Y ⊂ B, the following is true: |X||Y |(d(A, B) − d(X, Y)) = O(|A||B|). This definition was given by T.Tao [10]. Regularity is a key concept in Szemeredi's Regularity Lemma [´ 3]. In the standard definition, it is required that link density deviation is -small for all large subsets. Tao's definition suits us better since there is not such constraint on subset sizes. Small sets are regular Fig. 1.

A bipartite graph, where links can exist only between nodes in sets A and B; link density d is fraction of pairs (i, j) ∈ A × B that have links. in this sense, just because of their small sizes assuming large sets A and B. For each bipartite graph, define a function: L : 2A ×2 B → Q, L(X, Y) = |X||Y |(d(A, B)−d(X, Y)), in which 2M, denotes set of all subsets of M and Q is set of rational numbers. For better interpretation, we rewrite: L(X, Y) = |X||Y |d(A, B) − e(X, Y) = = Ede(X, Y) − e(X, Y), in which Ed denotes the expectation operator in a random bipartite graph with the link probability equal to d = d(A, B). As a result, L(X, Y) is simply deviation of the number of links in the subgraph, induced by X ∪ Y , from the expected number of links in the random bipartite graph. In this work we work with minimization of L. Maximization is done similarly using −L as the cost function. As a result, minL, corresponds finding the largest fluctuation that exceeds most the expected value Ee(·, ·). Quantum annealers, like D-Wave, are capable of solving quadratic binary optimization problems (qubo): min s X i,j (Ji,j sisj + hisi), (1) in which J and h are fixed matrix-valued parameters and s is a vector of binary variables. We can easily write the minimization of L in this form. For given subsets X and Y assign the values of binary variables si ∈ {0, 1} to all nodes in i ∈ V : i /∈ X ∪ Y ⇒ si = 0, i ∈ X ∪ Y ⇒ si = 1. As a result: |X| = X i∈A si , |Y | = X i∈B si . Similarly: e(X, Y) = X i∈A,j∈B ai,j sisj , in which (A)i,j = ai,j is the adjacency matrix of G.


Funding (F)
Output 1
Delivery 3a (D3a)
Bob can spend 0.6 BTC
from this output immediately
when C3a is broadcast
Output 0
Revocable Delivery 3a (RD3a)
Only Alice can broadcast 1000
confirmations from C3a's mined block
Alice & Bob can agree to create a spend
invalidation this with no time limitation
Output: Alice 0.4
1000 block Relative Confirmations Lock
Commitment 3a (C3a)
Only Alice can broadcast
Outputs:
0. RSMC Alice&Bob 0.4 BTC
1. Bob 0.6 BTC
No LockTime
Commitment 3b (C3b)
Only Bob can broadcast
Outputs:
0. Alice 0.4 BTC
1. RSMC Alice&Bob 0.6 BTC
No LockTime
Output 0
Delivery (D3b)
Alice can spend 0.4 BTC
from this output immediately
when C3b is broadcast
Output 1
Revocable Delivery 3b (RD3b)
Only Bob can broadcast 1000
confirmations from C3b's mined block
Alice & Bob can agree to create a spend
invalidating this with no time limitation
Output: Bob 0.6
1000 block Relative Confirmations Lock

Using these notations and because by definition |X||Y |d(X, Y) = e(X, Y), we can write the above program as a qubo: (X∗ 1 , X∗ 2) = arg min X,Y X i∈V1,j∈V2 (d(A, B) − ai,j)sisj , (2) X = {i ∈ A : si = 1}, Y = {j ∈ B : sj = 1}) Going trough all the configurations of s-variables is equivalent of going trough all subsets X and Y . Define the following block matrix M: {i, j} ⊂ V1 or {i, j} ⊂ V2 =⇒ (M)i,j = 0 and otherwise (M)i,j = d(A, B) − ai,j . Using M we can write: L(X, Y) = 1 2 X i,j siMi,j sj = 1 2 (s, Ms), (3) in which s is the vector of s-variables, (·, ·) is inner product of vectors and the summing is over all indices i and j. -regularity of a bipartite graph means that L is -bounded function for that graph.

As a result, finding global minimum and maximum of this function would resolve the - regularity check decision problem. Since this problem is coNP-complete, it is likely that there are no efficient algorithms for finding the minimum and maximum of L function for all graphs. For this reason, the optimization of L can provide a needed challenge for quantum computing to demonstrate its power.



III. COMMUNITY DETECTION ALGORITHM

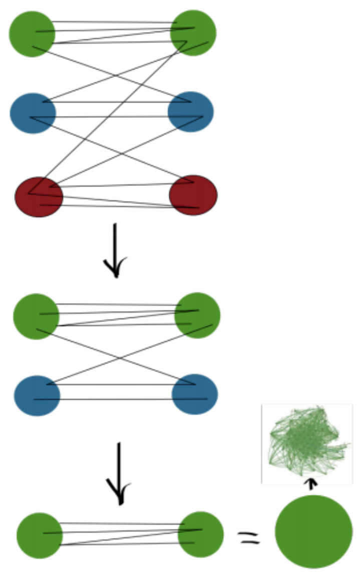
A. Stochastic block model

As we see in the following, finding the minL in a bipartite graph can be seen also as a basic operation in finding communities in a graph. We consider the case when the graph has communities generated from a stochastic block model (SBM), for a review see [2]. SBM(n, k, P, D) is a generative probabilistic graph model defined as follows. Here P is a probability distribution on [k] = {1, . . . , k} and D is a symmetric k-by-k matrix with entries Di,j ∈ [0, 1]. The model is generated by first sampling node labels $\sigma(1), \ldots, \sigma(n)$ independently from P, and then creating a random graph on node the set V by linking each unordered node pair $\{u, v\}$ with probability $D_{\sigma(u),\sigma(v)}$, independently of other node pairs. The node labeling $\sigma : V \to [k]$ partitions the node set into k disjoint communities $V_i = \sigma^{-1}(i)$, so that $V = V_1 \cup \cdots V_k$. Conditionally on the node labeling σ, the nodes between communities V_i and V_j are hence linked with probability $D_{i,j}$. The resulting random graph is denoted as G(n, k, P, D). Fig. 2. Generation of a bipartite graph G0 by a random split. At the top is a generic graph with communities indicated by different colors. Nodes are divided into two random sets, say, by tossing a fair coin. Each community is roughly split into two parts, one at the left (A) and other in the right (B). For three communities (indicated by green, blue and red balls) the bipartite graph G0 is shown in the lower part. Only those links in G joining A and B are preserved in G0 .

B. Community detection algorithm
In our previous works, we have extensively referred to SRL as a basis for graph analysis using various SBMs as a modeling space [1], [5]–[9]. Here we introduce another contact point between SBM and SRL. Assume that a graph G is drawn from G(n, k, P, D) as described above. We also assume that n is large enough. The first step is to find a bipartite subgraph, G0 of G: • divide nodes of G into two disjoint sets A and B, by tossing a fair coin for each node • G0 inherits all links from G that join A and B while all links inside A and B are deleted. This procedure is schematically shown in Fig. 2 We denote: $A_i := A \cap V_i$ and $B_i := B \cap V_i$ with sizes $a_i := |A_i|$ and $b_i := |B_i|$ for $i = 1, \cdots, k$. It is clear that random variables $(a_1, \cdots, a_k, b_1, \cdots, b_k)$ have a multinomial distribution with expectations $Ea_i = Eb_i = nP_i/2$ for all i. For large n these random variables are well concentrated around their expected values. Denote by d the link density of bipartite graph: $d = e(A, B)/(|A||B|)$. For a large graph, d is close to the expected link density of the original graph G, d(G), with high probability. We require the following $d(G) - D_{i,i} < 0, \forall i$. (4) This inequality means that all communities have internal density above the average density. For large graphs, we also have with high probability: $d - D_{i,i} < 0, \forall i$. It is required that the Condition 4 holds when G is replaced by a subgraph of G in which arbitrary communities are deleted. Fig. 3. The stages of community detection algorithm on a bipartite graph with split communities indicated by colors. At each stage, some communities are deleted. The program ends when only one community is left, the green ball in the figure. By deleting the found community from the whole graph, one could proceed to find the next community and so on until all are found. We do not provide a lengthy proof of the last claim. Typically probabilistic estimates are exponential, so this statement has high probability already with moderate graph sizes.

The idea behind community detection is the following. If there are bigger densities of links inside the communities than those between the communities, then the communities in the split graph are associated with the denser parts of the corresponding bipartite graph. As a result, there is a chance that communities can be found with the help of arg minL applied to the split graph. The next step of the algorithm is to construct the L-function for the bipartite graph of the split communities described above. The output of the algorithm is the subgraph induced by arg minL. The algorithm works correctly if the following conjecture is true: Conjecture 1. Consider graph G that is generated from G(n, k, P, D) with communities $V_1, \cdots, V_k$ and Condition 4 holds. Construct a random evenly split bipartite graph G0 with bipartition (A, B), $A = \cup_i A_i$ and $B = \cup_i B_i$ and in which sets with indices i are subsets of community V_i for $i = 1, \cdots, k$. Let $L(\cdot, \cdot)$ correspond to graph G0 . Then with probability tending to 1 as $1 - \exp(-n z)$ when $n \to \infty$ and with some constant $z > 0$, the following holds: $(X^*, Y^*) = \arg\min L(X, Y) \Rightarrow$ there is a proper subset of indices $I \subset \{1, \cdots, k\}$ such that $X^* = \cup_{i \in I} A_i$ and $Y^* = \cup_{i \in I} B_i$. We do not possess a full proof of this claim. As a first sketch, we consider optimization of expected L-function conditional to the sizes of split communities. The basic setting is the same as in Conjecture 1. We denote $x_i = |X \cap A_i|$ and $y_i = |Y \cap B_i|$. These integers are bounded by 0 ≤ xi ≤ ai and 0 ≤ yi ≤ bi . In shorthand, we write x ∈ [0, a] and y ∈ [0, b] and recall that such constraints are usually referred as box constraints [12]. In these notations, we have: L(X, Y) = X i,j (xiyjd(G 0) − e(X ∩ Ai , Y ∩ Bj)). Let us condition with respect to a1, · · · , ak, b1, · · · bk and take the expectation of L over the SBM: L1(X, Y) := EL(X, Y) = X i,j xixj (d − Di,j), in which d is the expected link density of the bipartite graph conditionally on the underlying community structure. We assume that we are in the high-probability event when the Condition 4 holds. Proposition 2. (X∗ , Y ∗) = arg min L1(X, Y) has the same structure as in Conjecture 1 in the sense that for the (X∗ , Y ∗), (xi , yi) ∈ {(0, 0),(ai , bi)} for all i, and there exists index i such that (xi , yi) = (0, 0). Proof. (A sketch) Let us consider a relaxation of the optimization problem in which the integer variables xi and yi are replaced by their continuous counterparts, keeping the box constraints.

We use the same symbols xi and yi to correspond to the global minimum of L1 within the box constraints and in the continuous variables. Denote L1(X∗ , Y ∗) = X i,j ximi,jyj . There is no global optimum strictly inside the box. To see why, denote the partial derivatives of L1 by fi := ∂xiL1(X∗ , Y ∗) and gi := ∂yiL1(X∗ , Y ∗). A global optimum inside the box would lead to L1(X∗ , Y ∗) = X i fixi = 0, which is not the global minimum since L1 can take negative values, say, when we take just one community, L1(A1, B1) = a1(d − D1,1)b1 < 0 by Condition 4. That is why the optimal point is on the boundary of the box. It must also be in the ’corners’ of the box, meaning that components have values 0 or have the maximal possible value. In a point that is on the box boundary but not at a corner point, the gradient of L1 is pointing inside the box volume and by moving towards some direction, provided the gradient is not perpendicular to the boundary, one could reduce the value of L1, this is impossible only if the point is in one of the corners of the box. If the gradient is perpendicular to boundary of the box, we would have (∇xL1)i = cδα,i and (∇yL1)i = c 0 δβ,i and L1 = (x, ∇xL1) = xαc P i (d − Dα,i)yi = yβc 0 P i (d − Dβ,i)xi in that point. As a result c = c 0 and α = β and L1 = xαyβ(d − Dα,α) ≥ nαmα(d − Dα,α). The found lower bound corresponds to a choice of a corner point and thus such a solution has lower energy than the suggested orthogonal to the boundary of the box. It is also easy to see that if xi = ai , then also yi = bi . This is due to ai ≈ bi and Condition 4.



As a result, the global optimum is just one of the corner points. The corner points of the box have integer coordinates and as a result the found minimum is a solution of the original integer problem. To prove Conjecture 1, we need some probability concentration inequalities like Chernoff bounds for known distributions with which we are dealing. The martingale argument may be used as was shown in an analogous case [11].

Proposition 2 shows that the Conjecture holds on average and provides a starting point for the proof. Then one should use concentration inequalities of probability theory to show that the solution of stochastic problem is the same with high probability. After the first step, the algorithm proceeds similarly on the found subgraph corresponding to arg minL. It runs until only one community is left. Then the found community is deleted from G and the whole process is repeated until all communities are found, see Fig 3.

IV. SIMULATIONS AND EXPERIMENTS WITH D-WAVE MACHINE

A. Brief description of the D-Wave Leap system

D-Wave Systems Inc. has published a quantum computing cloud service, Leap [13], for free trial and an option for buying quantum processing time. The Leap provides immediate access to a D-Wave 2000Q quantum computer or annealer. The computer has up to 2048 qubits and the service provides support for users such as the demos, interactive learning material and the Ocean software development kit (SDK) with suite of open-source Python tools and templates. When one has a qubo (2) in the form of (3), it is quite straightforward to implement and run it on the quantum computer. The size of the problem is restricted by the connectivity between qubits in the D-wave 2000Q Chimera architecture and the number of qubits available. For an arbitrary problem (2) an embedding is needed and that can drastically reduce the size of the problem that can be solved. For large problems, a hybrid approach is suggested, in which the problem is split into smaller pieces and part of the computations are done classically, so-called qbsolver. The Ocean SDK provides functions for automatic embedding as well as for solving a large qubo by qbsolver. D-Wave has announced that the next generation version of the quantum computer with over 5000 qubits and added connectivity between qubits would be available in mid-2020.

B. Experiments

1) Regularity check of a cortical area graph: Our first example is a small bipartite graph in Fig. 4, with 18 nodes taken from [5], in which SRL was used to analyse connections between cortical areas in a brain of a primate. This graph represents one regular pair of a bigger graph. Our task is to find a solution of the qubo (2) for this graph which is equivalent to the regularity check. In this case, the number of possible subset pairs is just 2 18 = 262144. As a result, the full search is possible. D-Wave finds the solution in a default time (microsecond). The Result: D-Wave finds the exact solution in one run taking microsecond. Fig. 4. Adjacency matrix (Ad) and the bipartite graph 2) Regularity check of random bipartite graphs: Next we solved qubo (2) for a random bipartite graph. In first experiments both segments have 50 nodes. The links between node pairs were drawn independently at random with a fixed probability = 0.2. In this case, it would be expensive to find the global minimum by exhaustive search. However, using the mixedinteger linear programming (MILP) solver CPLEX 12.9 with the model described in [14], we established that the found solution is the global minimum. Interestingly,

McGeoch and Wang [15] compared the CPLEX and the D-Wave machine Vesuvius 5 with 439 qubits on randomly generated Ising qubo instances. The Ising model was generated on a Chimera subgraph, so the J-matrix in qubo (1) was a weighted adjacency matrix of the Chimera subgraph. Such a problem is a sparse one. Dash [14] noted that with a suitable MILP model, CPLEX found the solution to the McGeoch and Wang instances very quickly and in a comparable time with D-Wave: For example with 512 nodes, the average time was 0.19s. D-Wave was used in hybrid quantum-classical mode when a part of the problem was solved on an ordinary computer and only smaller sub-problems are solved on D-Wave, so called qbsolver algorithm. This allows treating optimization problems with much larger number of variables than the number of qbits available in D-wave. It appears that our case of a dense bipartite graph is much harder than the problems that McGeoch, Wang and Dash studied. In the described 50×50 node graph the required time to find the optimal solution was 4.5 hours and to verify that the solution was the global minimum, took an additional 16 hours. The computer had four 2.7 GHz Xeon E5-4650 CPUs, with a total of 32 cores and 512 GB RAM.

The Result:
Both D-Wave and simulated annealing algorithm produce the same least energy solution when around 1000 instances were examined. This solution is the global minimum found with CPLEX. The required time at D-wave is again very small, less than a second, if the queuing time Fig. 5. Adjacency matrix of a bipartite graph Fig. 6. Adjacency matrix of largest irregularity pair of size 31 × 24 of the bipartite graph in Fig. 5 to the Leap cloud service is neglected. Simulated annealing needed around one minute. The sample graph had a density around 0.2044 and the found subgraph that is the solution of qubo (1) had density around 0.313. The corresponding non-zero blocks of adjacency matrices are shown in Figures 5-6. 3) Execution times on bipartite graphs: We tested solving the qubo (1), using large random bipartite graphs. For instance, in the case of 400 nodes it is not possible to use D-Wave in a similar way as in the case of 100 nodes. Instead, we used the simulated annealing algorithm, qbsolve with D-Wave or simulated annealing provided by the Leap system. The last two methods means that the problem is split into smaller pieces and the pieces are solved by classical or quantum annealing. In this way lager problems can be solved on D-Wave machine. For such scales of random dense graphs with several hundreds of nodes CPLEX becomes also impractical, due to long execution time. Simulated annealing also slows down.

For Fig. 7. Execution time in seconds of different optimization methods as a function of graph size in log-log scale. CPLEX can find exact solutions for small graphs and a poor solution for large graphs. In case of qbsolver with D-Wave annealer, the queuing time for cloud service is not filtered away. 2000 nodes, the time to find approximate solution is several hours on a laptop. In this case, it is not possible to verify with CPLEX whether a global minimum was found. As a results, such execution times are only lower bounds of the optimization time. The result is shown in Fig. 7.

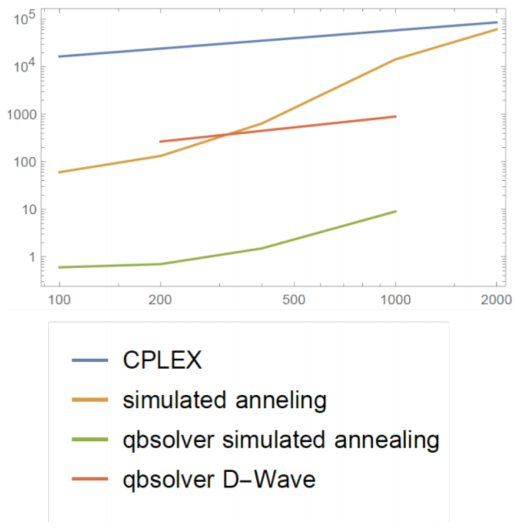


Fig. 7. Execution time in seconds of different optimization methods as a function of graph size in log-log scale. CPLEX can find exact solutions for small graphs and a poor solution for large graphs. In case of qbsolver with D-Wave annealer, the queuing time for cloud service is not filtered away.

The heuristical qbsolver classical annealer is the quickest, however producing slightly lower quality solutions for large graphs than the usual simulated annealer. The D-Wave qbsolver shows almost a constant time and, in this case, the queuing time is not filtered away. This may suggest that for extremely large cases, D-wave-assisted qbsolver is the winner in terms of time and quality. CPLEX can find exact solutions for small scales, but it is very slow and produces poor quality solutions for large graphs. We hypothesize that the qubo (1) for regularity check of a random bipartite graph can be a hard problem to solve already for moderate sizes of the underlying graph and can be used for testing quantum annealers. On the other hand, it suggests even some simple optimization problems emerging from large data can be only solved exactly with future quantum computers. 4) Small scale community detection: We tested our community detection algorithm using D-Wave and simulated annealing for a bipartite graph with 100 nodes and two communities. The adjacency matrix and the graph are shown in Figs. 8-9 The Result: both quantum - and classical annealers found the correct communities with 100 percent accuracy. Next experiment was done using only classical annealer because the graph had 200 nodes which cannot be embedded directly in the Chimera graph of D-Wave. The graph has three communities, see Fig. 10. In the first round, the largest and Fig. 8.

Adjacency matrix of a 100 node bipartite graph with two communities, seen as blocks of higher concentration of 1s Fig. 9. Bipartite graph with two communities, the denser community is at the right end of the plot sparsest community was dropped out. At the second round, one of the communities was left alone. As a result the algorithm found all three communities perfectly. 5) Towards large scale community detection: We consider a larger case of graph that has 2000 nodes and 10 communities. The adjacency matrix of the split bipartite graph is shown at the top of Fig. 11. The diagonal blocks of the two non-zero large blocks corresponds to links inside the communities. The darker color indicate higher density of links. In this case, the algorithm works as stated in Conjecture 1, the output is one community. Fig. 10. From upper left corner: 3 communities with internal links inside green boxes. The first round of iterations selects two most dense communities shown in black boxes. The next round picks one of the communities as an output. Result: perfect detection of all 3 communities Fig. 11. Community detection with a classical annealer for a graph with 10 communities and 2000 nodes. Top: adjacency graph of the bipartite graph with communities. Lower row, from left to right, the stages of community elimination. The number of communities at different stages are 10, 5, 3, 2 and 1. The last remaining community is the densest one. 1 2 3 4 5 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 Fig. 12. Density of the subgraphs at stages 1−5 of the community detection algorithm for the case shown in Fig. 11. The left-most dot indicates the density of the original graph = 0.146 · · · . The density shows a steady growth until only one community is left, at step 5. In our example, the last community is the densest of them, with a density of 0.680 · · · . This circumstance could be used as a simple criteria of stopping the algorithm, since after one community is left a further increase of the density is expected to be very small.

V. ALGORITHMS

We call our graph community detection algorithm as community panning. In this section we further scrutinize its details. The logical structure is given in the following Algorithm 1. The algorithm starts from a uniformly at random bipartitioning of the input graph. The follows the steps of finding maximally dense subgraph in the sense of regularity check, as described in previous sections. Obviously there is a problem of stopping. We suggest to use the cost function divided by the product of sizes of bi-partitions. This can be called energy per node. We claim that such a function has minimum at the right step of the algorithm. In our experiments this suggestion works well, see Fig. 13.

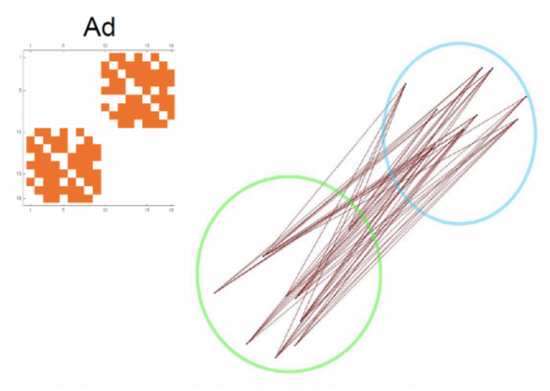


Fig. 4. Adjacency matrix (Ad) and the bipartite graph

Algorithm 1 Community panning algorithm 1: procedure FIND GRAPH COMMUNITIES(G) . Graph G 2: Read adjacency matrix A of G 3: Divide nodes (V) of G in two random sets V 1 and V 2 4: Find non-zero block B(V 1, V 2) of the adjacency matrix of the bipartite graph induced by V 1 and V 2 5: set e1 = 0, e2 = 0 6: while e1 > e2 do . Stop at the minimum of energy per node 7: e1 = e2 8: Find non-zero block B(V 1, V 2) of the adjacency matrix of the bipartite graph induced by V 1 and V 2, find link density of B, d 9: define qubo: bqm = P i∈V 1,j∈V 2 si(d − Bi,j)sj 10: call D-Wave to find arg minsi,sj (bqm) = L, si ∈ {0, 1} 11: e = bqm(L) 12: n1 = |V 1|, n2 = |V 2|, L = L(s1, · · · , sn1+n2) 13: V 1 0 = V 1 V 2 0 = V 2 14: update: V 1 = {i : 1 ≤ i ≤ n1, si = 1} 15: update: V 2 = {i : n1 + 1 ≤ i ≤ n1 + n2, si = 1} 16: n1 = |V 1|, n2 = |V 2| 17: e2 = e/n1/n2 18: return V 1 0 and V 2 0 . V 1 0 ∪ V 2 0 is the list of nodes in the community We made Python version of the corresponding algorithm available at the GitHub, [18]. It can be used with D-Wave or without it implementing a version with classical annealing. The latter version is quite quick for moderate size graphs like the one in Fig. 13, with 250 and five communities.

Next we implemented an algorithm that loops the first algorithm to find all communities. In this case, there is a problem how to stop algorithm or in other words, how to decide when all communities are found. We suggest to use adjacency matrix visualisation or cost function plotting as way, see Fig. 14 for details. VI. CONCLUSION Large graphs emerge from big data analysis and can pose serious computational problems. Szemeredi's Regularity Lemma ´ (SRL) is a fundamental tool in large graph analysis and thus can become important also in the big data area. Quantum computing is a new emergent area in computation and can contribute in both areas. We demonstrated connections between SRL, graph community detection and quantum annealing. SRL contains a very hard problem, regularity check, that could be solved on a quantum annealer, which we demonstrated using D-Wave quantum annealer. In case of graph community detection, we conjecture that quantum annealers of the future can produce high quality solutions for large scale problems. Fig. 13. The result of using community panning for a graph. Top-left adjacency matrix of the graph with communities. Yellow colour indicates value 1 and dark colour 0. The nodes are ordered in such a way that communities are apparent as diagonal blocks.

From the adjacency matrix a random bipartite graph is chosen. Its non-zero block of the adjacency matrix is at the top right. First round of the algorithm chooses the corresponding subgraph with adjacency block shown at the middle left. Apparently it contains just two communities. Similarly the next step finds a community with the adjacency block in the middle right. It coincides perfectly with one of the original communities, corresponding to the most yellow diagonal block in the whole graph in the top left figure. At the bottom is the plot of energy per node function at each stage. The step t = 2 has minimum of this function. The subgraph corresponding to the minimum of the energy per node is the solution yielding one community. The other points in the energy plot show larger energy per node if the number of steps is too small or too large. Research and business are already joining efforts in quantum computing and addressing big data. In 2019, IT giant Google and German research center Forschungszentrum Julich an- ¨ nounced a research partnership to develop quantum computing technology [17].