Article

Not peer-reviewed version

Probabilistic Cellular Automata Monte Carlo for the Maximum Clique Problem

Alessio Troiani *

Posted Date: 30 August 2024

doi: 10.20944/preprints202408.2259.v1

Keywords: Maximum Clique Problem; Probabilistic Cellular Automata; Monte Carlo Markov Chain; QUBO; Parallel Computing

 doi:10.20944/preprints202408.2259.v1

Article

Probabilistic Cellular Automata Monte Carlo for the Maximum Clique Problem

Alessio Troiani

Dipartimento di Matematica e Informatica, Università di Perugia; alessio.troiani@unipg.it
* Correspondence: alessio.troiani@unipg.it

Abstract: We consider the problem of finding the largest clique of a graph. This is an NP-hard problem and no exact algorithm to solve it exactly in polynomial time is known to exist. Several heuristic approaches have been proposed to find approximate solutions, Markov Chain Monte Carlo being one of those. In the context of Markov Chain Monte Carlo, we present a class of "parallel dynamics", known as Probabilistic Cellular Automata, which can be used in place of the more standard choice of sequential "single spin flip" to sample from a probability distribution concentrated on the largest cliques of the graph. We perform a numerical comparison between the two classes of chains both in terms of the quality of the solution and in terms of computational time. We show that the parallel dynamics are considerably faster than the sequential one while providing solutions of comparable quality.

Keywords: maximum clique problem; probabilistic cellular automata; Monte Carlo Markov Chain; QUBO; parallel computing

1. Introduction

A graph is a combinatorial structure consisting of a set of objects called *vertices* or *nodes* together with a collection of links, called *edges*, between pairs of vertices. Graphs may be used to model the relationship between pairs of entities of a population such as those between individuals in a social network or between data points in a dataset.

A typical question in decision theory is: "Given a graph G with set of vertices V, is there a subset $A \subset V$ with cardinality k such that all pairs of vertices in A are connected by an edge?". This problem is known as the *clique* problem and is known to be NP-complete (it is, actually, one of the 21 problems identified by Karp in [1]).

Its formulation as an optimization problem "what is the largest clique of a given graph G" is known as the Maximum Clique Problem and is NP-hard.

Though it is easy to verify that a collection of k nodes is indeed a clique for a graph G (where "easy" means that this verification can be done in polynomial time - this is actually the definition of an NP problem), no polynomial time algorithm is known to exist to determine the size of the largest clique of a graph. For this reason, in many situations where computation time is limited, finding an optimal clique using an exact algorithm is usually not a viable option. In these cases, having an algorithm capable of finding a good heuristic solution in a short time may be extremely valuable.

Beyond the exact algorithms, which, in the general case, are not suitable for large instances, several approaches have been used to tackle the clique problem such as greedy procedures, Monte Carlo methods, machine learning and artificial intelligence tools, and quantum computing algorithms. A recent overview can be found here [2].

Here, our focus is on Monte Carlo methods. The idea behind Monte Carlo methods consists in the ability to sample collections of nodes from a probability distribution which puts most of the weight on the largest cliques of a given graph. This probability distribution is usually approximated by the long-term distribution of a suitable Markov Chain (see [3] for an elementary introduction).

In the context of the Maximum Clique problem, Markov Chain Monte Carlo (MCMC) methods have been analyzed extensively from both the theoretical and numerical point of view. Even though in

 doi:10.20944/preprints202408.2259.v1

[4] some of the limitations of MCMC have been highlighted, more recent results such as those in [5–7] showed that this approach may be worth tackling practical instances.

This work aims to evaluate, numerically, a class of Markov Chain Monte Carlo algorithms, which here we refer to as Probabilistic Cellular Automata Monte Carlo (PCAMC), where the Markov chain lives on some multidimensional discrete space and at each step, all the components of the "current state" are updated independently one of the other. This is in contrast with the usual choice in MCMC of updating only one component at each step of the dynamics. The inherent parallel nature of PCAMC allows for a straightforward implementation that takes full advantage of parallel computing architecture giving PCAMC a significant advantage over standard MCMC methods from the computational point of view.

These algorithms have already been considered (by the author of this paper and several coauthors) to tackle certain instances of the Quadratic Unconstrained Binary Optimization Problem (QUBO) and proved to be rather effective. Here we will use them in the context of the Maximum Clique Problem and we will show that their performances are quite promising.

2. Materials and Methods

In this section, we give a mathematical definition of the maximum clique problem and see how it can be formulated in terms of the minimization of a suitable objective function. We, then, describe the Monte Carlo algorithms we want to evaluate, namely the one we refer to as the PCA (and that throughout the paper we denote by $\mathcal{A}$) and the Shaken dynamics (denoted by $\mathcal{S}$) and the algorithm we take as a reference, that is, the Metropolis "single spin flip" algorithm (that we denote by $\mathcal{M}$).

2.1. Preliminary Definitions and Statement of the Problem

Let $G = (V, E)$ be a non-oriented graph where V is the set of vertices and $E \subset V \times V$ is the set of edges. A graph G is said to be *complete* if $E = \{\{i, j\}$ for all $i, j \in V, i \neq j\}$, that is if there as an edge between each pair of vertices of G. A graph $G' = (V', E')$ is called a *subgraph* of G if $V' \subset V$ and $E' \subset E$. Let $A \subset V$ be a subset of the vertices of G. Then the graph $G[A]$ with set of vertices A and set of edges $E|_A = \{\{i, j\} : i, j \in A\}$ (that is the set of edges of G with both endpoints in A) is called the subgraph of G *induced* by A. The subset $C \subset V$ is called a *clique* for G if the induced subgraph $G[C]$ is complete. We denote by $\mathcal{K}(\mathcal{G})$ the set of all cliques of G. A clique C is said to be *maximal* for G if, for all $C' \subset V$ such that $C' \supset C$, C' is not a clique. In words, this means that C is maximal if it can not be extended by adding to C an adjacent vertex. Further, a clique C is called a *maximum clique* of G if $|C| \geq |C'|$ for all $C' in \mathcal{K}(\mathcal{G})$ where $|S|$ denotes the cardinality of set $|S|$. The cadinalitiy of a maximum clique of G is called the *clique number* of G and is denoted by $\omega(G)\ldots$

Given a graph G, the *maximum clique problem* consists in determining the clique number $\omega(G)$, that is, the size of the largest clique of the graph. As mentioned in the introduction, this problem is NP-hard.

A graph $G = (V, E)$ can be encoded in several ways. One possibility is that of considering its *adjacency matrix* $A_G = (a_{ij})_{i,j=1\ldots N}$ defined by $a_{ij} = 1$ if $\{i, j\} \in E$ and $a_{ij} = 0$ otherwise. Note that A_G is a symmetric matrix since G is non-oriented.

2.2. A Hamiltonian for the Clique Problem

A discrete optimization problem consists in finding the minimum of an objective function $H(\eta)$ where η is an element of a suitable multidimensional space $\mathcal{X}$.

Consider a graph $G = (V, E)$, with vertices $V = \{1, 2, \ldots, N\}$ and the space of *boolean configurations* with N components $\mathcal{X} = \{0, 1\}^N$. Then, for each subgraph $G[A]$, with $A \subset V$, there is one, and only one, $\eta^A \in \mathcal{X}$ defined by $\eta^A_i = 1 \iff i \in A$. We call the set of indices where η is positive the support of η end denoted by $\text{supp}(\eta)$ (in words $\text{supp}(\eta)$ is the set of vertices selected by the configuration η). We can, therefore, identify subgraphs of a graph with N vertices and configurations of $\{0, 1\}^N$ and. With an abuse of notation we write "the subgraph η'' in place of the subgraph $G[A]$

 doi:10.20944/preprints202408.2259.v1

induced by the set $A = \{i \in 1, \dots, N : \eta_i = 1\}$. Similarly, if the vertices of subgraph η are a clique of G we refer to η as a clique of G. Moreover we write $|\eta|$ for the cardinalitiy of a configuration η defined as $|\eta| = |\{i \in 1, \dots, N : \eta_i = 1\}|$.

For any graph $G = (V, E)$ with N vertices it is possible to define the $N \times N$ matrix $J = J(G)$ as $J_{ij} = \frac{1}{2}$ if $\{i, j\} \notin E$ and $J_{ij} = 0$ otherwise. We call the matrix $J(G)$ defined in this way the matrix of *missing edges* of G. Note that the matrix J is symmetric and is closely related to the adjacency matrix of the graph. In particular, if $A = A(G)$ is the adjacency matrix of the graph, then $J_{ij} = \frac{1}{2}(1 - A_{ij})$ if $i \neq j$ and 0 otherwise. Therefore, any symmetric matrix J with zeroes on the diagonal and entries in $\left\{0, \frac{1}{2}\right\}$ uniquely determines a graph $G(J)$. .Moreover, when it does not give rise to confusion we drop the explicit dependence of J on G.

Given a graph $G = (V, E)$ consider the associated matrix of missing edges J and define the function

$$H^{J,\lambda}(\eta) = \sum_{i,j \leq N} J_{ij}\eta_i\eta_j - \lambda \sum_{i=1}^{N} \eta_i. \tag{1}$$

In the remainder of the paper, to lighten the notation, we will write H instead of $H^{J,\lambda}$.

Lemma 1. *Let $G = (V, E)$ be a non-oriented graph. If $0 < \lambda < 1$ then the H is minimal if and only if η is a maximum clique of G.*

Proof. Let us first show that the condition $0 < \lambda < 1$ is sufficient.

Let η_k be a configuration with cardinality k. Then $H(\eta_k) = -\lambda k$ if η_k and $H(\eta_k) > -\lambda k$ otherwise since the term $\sum_{i,j \leq N} J_{ij}\eta_i\eta_j > 0$ for all η's that are not cliques (since there is at least one "missing edge").

Let $\omega(G)$ be the clique number of G, that is, the cardinality of the largest clique of G. Then there exists a clique $\bar{\eta}$ with cardinality $\omega(G)$such that $H(\bar{\eta}) = -\lambda\omega(G)$. From the previous observation, it follows that $H(\eta) \geq -\lambda\omega(G)$ for all η's with cardinality at most $\omega(G)$ with equality only if η is a maximum clique for G.

It is left to show that, if the cardinality of η is larger than $\omega(G)$, then $H(\eta) > -\lambda\omega(G)$. Let η have cardinality $k > \omega(G)$. Write $S = \text{supp}(eta)$ and denote by A the maximum clique of $G[S]$. Further let $B = S \setminus A$ and call η^A and η^B the configurations whose support is, respctively, A and B. Note that A and B are disjoint by construction and $|B| \geq 1$ since, by assumption $|A| + |B| > \omega(G)$ and $|A| \leq \omega(G)$. We have

$$H(\eta) = H(\eta^A) + H(\eta^B) + \sum_{i \in A, j \in B} J_{ij}\eta_i\eta_j. \tag{2}$$

Now observe that, by the assumed maximality of clique A, $\sum_{i \in A, j \in B} J_{ij}\eta_i\eta_j \geq |B|$ since there must be at least one missing edge between each vertex in B and the vertices of A. Moreover, it must be $H(\eta^A) \geq -\lambda\omega(G)$ and $H(\eta^B) \geq -\lambda|B|$. Hence

$$H(\eta) \geq -\lambda\omega(G) + (1 - \lambda)|B| \geq -\lambda\omega(G) \tag{3}$$

as soon as $\lambda < 1$.

To see that the condition $\lambda < 1$ is necessary, consider the case where η is a clique of size k. Then $H(\eta) = -\lambda k$. Let A be the set of indices such that $\eta_i = 1$ (that is, the set of vertices of the clique). Consider the configuration τ obtained from η by setting $\eta_j = 1$ for some $j \notin A$ and suppose that there is an edge between vertex j and $k - 1$ vertices in A. Then $H(\tau) = -\lambda(k + 1) + 1$. Then $H(\tau) \leq H(\eta)$ as soon as $\lambda >= 1$.

 doi:10.20944/preprints202408.2259.v1

Finally observe that if $\lambda = 0$, $H(\eta) = 0$ for all η's that are a clique and if $\lambda < 0$ only the empty configuration $\vec{0}$ has energy $H(\vec{0}) = 0$ whereas $H(\eta) > 0$ for all $\eta \neq \vec{0}$.

□

Thanks to the previous lemma, the maximum clique problem can be formulated in a standard optimization problem form as

$$\min_{\eta\in\{0,1\}^N} H(\eta) = \min_{\eta\in\{0,1\}^N} \sum_{i,j\leq N} J_{ij}\eta_i\eta_j - \lambda \sum_{i=1}^{N} \eta_i, \qquad 0 < \lambda < 1. \tag{4}$$

Note that, exploiting the fact that $\eta_i = \eta_i^2$, It is possible to rewrite the previous formula in terms of a matrix $\tilde{J} = \tilde{J}(\lambda)$ as

$$\min_{\eta\in\{0,1\}^N} H(\eta) = \min_{\eta\in\{0,1\}^N} \sum_{ij\leq N} \tilde{J}_{ij}\eta_i\eta_j = \min_{\eta\in\{0,1\}^N} \eta^T \tilde{J}\eta \tag{5}$$

with $\tilde{J}_{ij} = J_{ij}$ if $i \neq j$ and $\tilde{J}_{ii} = -\lambda$. This formulation is known in the literature as Quadratic Unconstrained Binary Optimization Problem (QUBO): an NP-hard problem with a vast range of applications (see the text [8] for an overview) that allows for a straightforward embedding of (beyond the maximum clique problem) of several combinatorial optimization problems such as max-cut and graph coloring (see [9]. Further, QUBO is a model of interest in the field of quantum computing (see, e.g., [10–12]),

It is worth noting that the objective function of an optimization problem can be interpreted as the Hamiltonian energy function of a statistical mechanics model with state space $\mathcal{X}$. The statistical mechanics model's ground states (minima of the Hamiltonian) correspond to configurations minimizing the objective function. The equilibrium distribution of a statistical mechanics model with Hamiltonian H is described by the Gibbs measure

$$\mu_G(\eta) = \frac{1}{Z} e^{-\beta H(\eta)}, \quad \eta \in \mathcal{X} \tag{6}$$

where β is a positive real parameter called the inverse temperature and Z is a normalizing constant called the partition function. As $\beta \to \infty$ (low-temperature regime) this probability measure concentrates on the ground states of the system. This fact suggests that a solution to the optimization problem can be found by sampling from the Gibbs measure for the corresponding statistical mechanics model at low temperature.

To this aim, it is possible to define a Markov chain on $\mathcal{X}$ with transition probability $P : \mathcal{X} \times \mathcal{X} \to [0, 1]$ whose stationary distribution is the Gibbs measure μ_g. If the chain is irreducible and aperiodic, then, irrespective of the initial starting configuration, the probability distribution of the chain after n steps tends to the stationary distribution as $n \to \infty$. The sampling from the Gibbs measure can, then, be performed by letting the chain evolve for a *sufficiently long* time. More precisely, let $\mu_n(\eta)$ be the probability distribution after n steps of the chain starting from configuration η and let π be the stationary measure for the chain with transition probability P. Then, denoting by $\mathrm{d_{TV}}(\cdot, \cdot)$ the total variation distance.

$$\mathrm{d_{TV}}(\mu_n(\eta), \pi) \to 0 \text{ as } n \to \infty. \tag{7}$$

The time it takes for the chain to have a distribution that is within a distance ε from the equilibrium distribution is referred to as the *mixing time* of the chain. In formulae, the mixing time of a Markov Chain with state space $\mathcal{X}$ and stationary distribution π is defined, for $\varepsilon < \frac{1}{2}$, as

$$t_{\mathrm{mix}}(\varepsilon) = \min\{n : \mathrm{d_{TV}}(\mu_n(\eta), \pi) \leq \varepsilon\} \tag{8}$$

 doi:10.20944/preprints202408.2259.v1

In general, determining a priori bound on the mixing time of the chain that guarantees that the algorithm is useful in practice (that is it has a mixing time that is at most polynomial in the size of the problem) may be non-trivial. Actually, in the case of Erdős-Rényi random graphs, Jerrum ([4]) proved that the mixing time of the Metropolis process he considered for the Maximum Clique Problem has super-polynomial mixing time. However, for instances that are not too big (number of nodes up to a few thousand) Monte Carlo algorithms may still provide some good heuristic solutions (see, e.g., [5].

2.3. The Metropolis Single Spin Flip Algorithm

A reference choice among Markov Chain Monte Carlo algorithms is the Metropolis update rule with single *spin flip* whose transition probability matrix is as follows.

For a given graph $G = (V, E)$ with missing edges matrix J, consider the Hamiltonian function H defined in (1). Let $\eta^{(i)}$ be the configuration obtained from η by flipping the occupation number of the i-th component, that is $\eta_i^{(i)} = 1 - \eta_i$ and $\eta_j^{(i)} = \eta_j$ for all $j \neq i$.

$$P_{\mathcal{M}}(\eta, \tau) = \begin{cases} \frac{1}{\mathrm{N}} e^{-\beta[H(\tau)-H(\eta)]_+}. & \text{if} \tau = \eta^{(i)}, i = 1 \dots N \\ 1 - \sum_i \frac{1}{\mathrm{N}} e^{-\beta[H(\eta^{(i)})-H(\eta)]_+} & \text{if} \tau = \eta \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $[\cdot]_+$ denotes the positive part. In words, at each step, an index $i \in 1, \dots, N$ is chosen a random and if the configuration obtained by *flipping* the value of the i-th component of η has a Hamiltonian energy lower than that of η the occupation number of η_i is changed with probability one. If this is not the case, the occupation number of η_i is changed with a probability that is exponentially small in the difference between the energy of the *target* configuration and the energy of the *current* one.

Let $\pi_{\mathcal{M}}(\eta) = \frac{1}{Z} e^{-\beta H(\eta)}$. Then it is immediate to verify that $P_{\mathcal{M}}$ satisfies the detailed balance condition

$$\pi_{\mathcal{M}}(\eta) P_{\mathcal{M}}(\eta, \tau) = \pi_{\mathcal{M}}(\tau) P_{\mathcal{M}}(\tau, \eta)$$

ensuring that $\pi_{\mathcal{M}}$ is the stationary distribution of $P_{\mathcal{M}}$.

We remark that this algorithm is allowed to visit configurations that are not cliques of the graph G. However, in the limit $\beta \to \infty$, if $\lambda \propto \frac{1}{\beta}$ the algorithm is equivalent (see [5]) to the Metropolis algorithm considered by Jerrum in [4] which allows transitions only between cliques of G.

2.4. The "Pair Hamiltonian" Probabilistic Cellular Automton

A probabilistic cellular automaton on $\mathcal{X}$ is a Markov Chain whose transition probability matrix can be written, for any $\eta, \tau \in \mathcal{X}$ as

$$P_{\eta,\tau} = \prod_i P(\tau_i = \cdot | \eta). \tag{10}$$

In words, this means that at each step, the value of each component τ_i of the target configuration τ can be sampled independently of the values of all other $\tau_j, j \neq i$. From a computational point of view this fact is particularly appealing, since, in principle, the update value of each component of τ could be demanded to a dedicated computing *core* (see [13]). With the advent of massively parallel processors such as GPUs, this is an actual possibility even on consumer hardware for configurations with several hundreds of components.

In [14] the PCA Monte Carlo algorithm has been introduced to study the minima of QUBO. When used to approach the maximum clique problem the algorithm is as follows.

 doi:10.20944/preprints202408.2259.v1

Consider the *pair Hamiltonian* function defined on pairs of configurations

$$H(\eta,\tau) = -\left(\beta\sum_{i,j}\tilde{J}_{ij}\eta_i\tau_j + q\sum_i[\eta_i(1-\tau_i)+\tau_i(1-\eta_i)]\right), \tag{11}$$

where $\tilde{J}$ is the matrix used in (5), and set the transition probability to be

$$P_{\mathcal{A}}(\eta,\tau) = e^{-H(\eta,\tau)} = \frac{e^{-H(\eta,\tau)}}{\sum_\tau e^{-H(\eta,\tau)}} \tag{12}$$

Let $h_i(\eta) = \sum_j \tilde{J}_{ij}\eta_j$ be the *local energy field* felt by the i-th component of η. Then $H(\eta,\tau)$ can be rewritten as $H(\eta,\tau) = \beta\sum_i h_i(\eta)\tau_i + q([\eta_i(1-\tau_i)+\tau_i(1-\eta_i)])$. Consequently, the transition probabilities can be rewritten as

$$P_{\mathcal{A}}(\eta,\tau) = \prod_i \frac{e^{-\beta h_i(\eta)\tau_i - q[\eta_i(1-\tau_i)+\tau_i(1-\eta_i)]}}{Z_i} \tag{13}$$

yielding

$$\mathrm{P}(\tau_i = 1|\eta) = \frac{e^{-\beta h_i(\eta)-q(1-\eta_i)}}{Z_i} \quad \text{and} \quad \mathrm{P}(\tau_i = 0|\eta) = \frac{e^{-q\eta_i}}{Z_i} \tag{14}$$

with $Z_i = e^{-\beta h_i - q(1-\eta_i)} + e^{-q\eta_i}$. Therefore $P_{\mathcal{X}}$ defines, indeed, a PCA in the sense of (10).

By the symmetry of $\tilde{J}$, a straightforward computation shows that the detailed balance condition

$$P_{\mathcal{A}}(\eta,\tau)\frac{\sum_\tau e^{-H(\eta,\tau)}}{\sum_{\eta,\tau} e^{-H(\eta,\tau)}} = P_{\mathcal{A}}(\tau,\eta)\frac{\sum_\eta e^{-H(\tau,\eta)}}{\sum_{\eta,\tau} e^{-H(\eta,\tau)}} \tag{15}$$

holds and, hence,

$$\pi_{\mathcal{A}}(\eta) = \frac{\sum_\tau e^{-H(\eta,\tau)}}{\sum_{\eta,\tau} e^{-H(\eta,\tau)}} \tag{16}$$

is the stationary measure of $P_{\mathcal{A}}$. Note that also this algorithm may visit all configurations and not only those that are cliques.

A closer look at equation (13) shows that at each step, the dynamics tends to select, for the target configuration τ, the vertices connected to all vertices selected by configuration η. Indeed, for these components, the local energy field is negative (equal to $-\lambda$). On the other hand, the probability of selecting, in τ, components for which there is at least one missing edge with the components selected by η is exponentially small (since $\lambda < 1$). Further note that the term $e^{-q[\eta_i(1-\tau_i+\tau_i(1-\eta_i))]}$ reduces the probability that the occupation number of the i-th component of τ differs from the i-th component of η so that the probability that η and τ differ at too many sites stays small.

To get an idea of why this algorithm is expected to work, it is possible to argue as follows. At first observe that $H(\eta,\eta) = \beta H(\eta)$ with H defined as in LABEL_ABOVE. Further note that, as q gets large, the weight of pairs (η,τ) with $\eta \neq \tau$ in $\pi_{\mathcal{A}}(\eta)$ is exponentially depressed $\pi_{\mathcal{A}}(\eta)$ becomes closer to the Gibbs measure $\frac{e^{-\beta H(\eta)}}{Z}$ with Hamiltonian H and inverse temperature β.

In [14] and in [15] the authors considered QUBO instances where the coefficients of the matrix $\tilde{J}$ are realization of independent identically distributed random variables with expected value zero and showed that the PCA Monte Carlo performed quite well when compared to other heuristic techniques. Here, however, the elements of the matrix $\tilde{J}$ have a particular structure: they are negative on the diagonal and positive or null outside of it and it is not, a priori, obvious how this structure could have an impact on the practical performances of the algorithm.

 doi:10.20944/preprints202408.2259.v1

2.5. The Shaken Dynamics

Shaken dynamics have been introduced in [16,17] in the process of finding efficient algorithms to sample from the Gibbs measure on spin systems (see [18,19]) and extend the class of PCA with transition probabilities of type $P_{\mathcal{A}} = \frac{e^{-\beta H(\eta,\tau)}}{\sum_\tau e^{-\beta H(\eta,\tau)}}$ described above and defined in terms of a pair Hamiltonian.

The transition probabilities of the Shaken dynamics come as a combination of two steps and are of the following type:

$$P_{\mathcal{S}}(\eta,\tau) = \sum_\sigma \overleftarrow{P}(\eta,\sigma)\overrightarrow{P}(\sigma,\tau) \tag{17}$$

with

$$\overleftarrow{P}(\eta,\sigma) = \frac{e^{-\overleftarrow{H}(\eta,\sigma)}}{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}} \quad \text{and} \quad \overrightarrow{P}(\eta,\sigma) = \frac{e^{-\overrightarrow{H}(\eta,\sigma)}}{\sum_\sigma e^{-\overrightarrow{H}(\eta,\sigma)}}. \tag{18}$$

The two functions $\overleftarrow{H}$ and $\overrightarrow{H}$ appearing in the definition of the transition probabilities must be such that $\overleftarrow{H}(\eta,\sigma) = \overrightarrow{H}(\sigma,\eta)$. If this is the case, then it follows that

$$\pi_{\mathcal{S}}(\eta) = \frac{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}}{\sum_{\eta,\sigma} e^{-\overleftarrow{H}(\eta,\sigma)}} \tag{19}$$

is the stationary (reversible) measure for $P_{\mathcal{S}}$. To see this, observe that the detailed balance condition holds. Indeed, calling $Z_\eta = \sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}$ and $Z = \sum_{\eta,\sigma} e^{-\overleftarrow{H}(\eta,\sigma)}$, we have

$$\pi_{\mathcal{S}}(\eta)P_{\mathcal{S}}(\eta,\tau) = \frac{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}}{\sum_{\eta,\sigma} e^{-\overleftarrow{H}(\eta,\sigma)}} \sum_\sigma \frac{e^{-\overleftarrow{H}(\eta,\sigma)}}{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}} \frac{e^{-\overrightarrow{H}(\sigma,\tau)}}{\sum_\tau e^{-\overrightarrow{H}(\sigma,\tau)}} \tag{20}$$

$$= \frac{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}}{\sum_{\eta,\sigma} e^{-\overleftarrow{H}(\eta,\sigma)}} \sum_\sigma \frac{e^{-\overrightarrow{H}(\sigma,\eta)}}{\sum_\sigma e^{-\overleftarrow{H}(\eta,\sigma)}} \frac{e^{-\overleftarrow{H}(\tau,\sigma)}}{\sum_\tau e^{-\overrightarrow{H}(\sigma,\tau)}} \tag{21}$$

$$= \frac{\sum_\tau e^{-\overleftarrow{H}(\tau,\sigma)}}{\sum_{\tau,\sigma} e^{-\overleftarrow{H}(\tau,\sigma)}} \sum_\sigma \frac{e^{-\overleftarrow{H}(\tau,\sigma)}}{\sum_\sigma e^{-\overleftarrow{H}(\tau,\sigma)}} \frac{e^{-\overrightarrow{H}(\sigma,\eta)}}{\sum_\sigma e^{-\overrightarrow{H}(\sigma,\eta)}} = \pi_{\mathcal{S}}(\tau)P_{\mathcal{S}}(\tau,\sigma) \tag{22}$$

Having defined the Shaken dynamics, we can describe how it can be used as a heuristic algorithm for the maximum clique problem.

Consider a graph $G = (V,E)$, with $|V| = N$, its associated missing edge matrix and J and the corresponding matrix $\tilde{J} = \tilde{J}(\lambda)$ used to formulate the maximum clique problem as QUBO (as in (5)).

Let $\overleftarrow{J}$ be such that $\overleftarrow{J} + \overleftarrow{J}^T = \tilde{J}$ and define

$$\overleftarrow{H}(\eta,\sigma) = -\beta \sum_{i,j\le N} \overleftarrow{J}_{ij}\eta_i\sigma_j - q\sum_i [\eta_i(1-\sigma_i) + \sigma_i(1-\eta_i)] \tag{23}$$

$$\overrightarrow{H}(\eta,\sigma) = -\beta \sum_{i,j\le N} \overleftarrow{J}^T_{ij}\eta_i\sigma_j - q\sum_i [\eta_i(1-\sigma_i) + \sigma_i(1-\eta_i)]. \tag{24}$$

Then the condition $\overleftarrow{H}(\eta,\sigma) = \overrightarrow{H}(\sigma,\eta)$ is verified. Further, observe that

$$\overleftarrow{H}(\eta,\eta) = \eta^T\tilde{J}\eta = \eta^T(\overleftarrow{J} + \overleftarrow{J}^T)\eta = 2(\eta^T\overleftarrow{J}\,\eta) = \frac{1}{2}\beta H(\eta) \tag{25}$$

where $H(\eta)$ is the same the one defined in (5).

 doi:10.20944/preprints202408.2259.v1

Strictly speaking, the Shaken dynamics is not a PCA. However the transition probabilities of each half-step are of the form (13) where, instead of the vector of field h, appear, alternatively, the two vectors of fields $\overleftarrow{h} = \overleftarrow{J} \cdot \eta$ and $\overrightarrow{h} = \overleftarrow{J^T} \cdot \sigma$.

Also in this case, if the parameter q is large, the stationary measure $\pi_S(\eta)$ is close to the Gibbs measure $\frac{e^{-\beta H(\eta)}}{Z}$ (the factor $e^{-1/2}$ appears both at the numerator and at the denominator and, therefore, cancels out) and, hence, the Shaken dynamics provides a heuristic algorithm for the maximum clique problem. Also this Markov chain lives on the whole set $\mathcal{X}$ and not only on the subsets corresponding to cliques.

In [17] the Shaken dynamics has been used, in the context of spin systems, to find the minima of a QUBO-like problem defined on a lattice that is a problem where the interaction J_{ij} was different from zero only for pairs i, j satisfying a certain relation. In that scenario, the Shaken dynamics appeared to be rather effective. However, in the case of the maximum clique problem, besides the requirement for J to be symmetric, there is no restriction on which (and how many!) pairs J_{ij} can be different from zero as long as $i \neq j$. Therefore the effectiveness of this approach to the maximum clique problem has to be assessed.

3. Results

We tested our algorithms on a set of benchmark graphs commonly considered in the literature (DIMACS benchmarks). The graph considered have a different number of nodes N spanning from 125 to 4000. The graphs taken into account are of several types which can be retrieved from the graph id:

Erdős-Rényi graphs: graphs with a fixed number of vertices and edges selected independently at random with uniform probability. These are graphs with id of type `Cn.d` (where `n` is the number of vertices and `d/10` is the density (probability) of edges) and `DSJCn_d` (where `n` is the number of nodes and `d/10` is the density of edges)

Steiner triple graphs: with id `MANN_aXX` are graphs constructed to have a clique formulation of the Steiner Triple Problem

Brockington graphs: with id `brockN_d` where `N` is the number of vertices and `d` is a parameter

Sanchis graphs: with id `genn_p0.x_y` where `n` is the number of vertices, `p0.x` is the density of edges and `y` is the size of the planted clique

Hamming graphs: with id `hammingx-y` with parameters `x` and `y`

Keller graphs: with id `kellern` and parameters `n` = 4, 5, 6

P-hat graphs: with id `p-hatn-x` where `n` is the number of nodes and `x` is an identifier; graphs generated with p-hat have wider node degree spread and larger cliques than uniform graphs.

More details can be found in [20].

We formulated the problem as in (5) and fixed, for all graphs and all algorithms, $\lambda = 0.25$.

As far as the parameters β and q, since we do not have an apriori argument for an optimal choice, we tested several pairs of values. However, to determine the values of q to perform our test, we kept into account the size N (number of nodes) of the graph and chose values of q so as to have $e^{-q}N = c_q\sqrt{N}$ for several values of c_q as described below. The rationale for this choice is the fact that, for $\mathcal{A}$ and S, the term e^{-q} is proportional to the probability of "flipping" a component neglecting the contribution depending on the "energy field" for that component. In this way we intended to change, at each iteration, a number of component *of the order* $\sqrt{N}$.

The values we used are the following:

Metropolis ($\mathcal{M}$): β in the range $1, 1.5, 2, \ldots, 4$

PCA ($\mathcal{A}$): β in the range $1, 1.5, 2, \ldots 4$ and c_q in the set $\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, 2, 3, 4$

Shaken dynamics ($\mathcal{S}$): in the range $1, 2, \ldots 7$ and c_q in the set $\{\frac{1}{64}, \frac{1}{32}, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, \frac{1}{1}\}$

These sets of ranges for the parameters have been determined after some experimentation where we saw that certain ranges of parameters were not suitable for a given algorithm. In particular, we needed to find values of q big enough (that is c_q sufficiently small) for the dynamics to retain a few components.

 doi:10.20944/preprints202408.2259.v1

Note that not all pairs of parameters (β, q) that have been taken into account are effective for a given algorithm.

For the Shaken dynamics $\mathcal{S}$, we also had to make a choice for the matrix $\overleftarrow{J}$. We proceeded as follows. For each $i \in 1, \ldots, N$ we set

$$\overleftarrow{J}_{ij} = \begin{cases} J_{ij'} & \text{with } j' = ((j-1) \bmod N) + 1 \text{ for } j \in (i+1, i+2, \ldots i + \left\lceil \frac{N}{2} \right\rceil - 1) \\ \frac{J_{ij}}{2} & \text{for } j = i + \frac{N}{2} \text{ if N is even} \\ 0 & \text{otherwise} \end{cases} \tag{26}$$

Intuitively, for each row, the matrix $\overleftarrow{J}$ is obtained retaining the $\frac{N}{2}$ elements of $\tilde{J}$ "on the right" of the diagonal element. It is straightforward to verify that $\overleftarrow{J} + \overleftarrow{J^T} = \tilde{J}$.

This choice also provides a reason for why the values β used for the algorithm $\mathcal{S}$ are in a range that is, roughly, twice as wide then that used for $\mathcal{M}$ and A: the fields used to update the configuration at each "half-step" are proportional to $\overleftarrow{J} \cdot \eta$ whose components could be expected to be approximately one-half of the components of $J \cdot eta$ (which are the fields determining the transitions of $\mathcal{A}$).

Simulations were run for a fixed number of iterations and the largest clique found in each run of any of the algorithms was recorded. For each set of parameters (β for $\mathcal{M}$ and (β, q) for $\mathcal{A}$ and $\mathcal{S}$) we run 10 simulations. The number of iterations has been set in the following way:

Metropolis ($\mathcal{M}$): 100000 *sweeps* where a sweep corresponds to N steps of the Markov Chain (that is a total of $100000N$ "attempted" flips);
PCA ($\mathcal{A}$): 200000 steps of the Markov Chain;
Shaken dynamics ($\mathcal{S}$): 100000 complete steps of the Markov Chain (200000 half-steps).

Our intent was to perform a fair comparison of the three algorithms. However it is not obvious what the right criterion to establish fairness should be. For instance, one could run the chains for the same number of "nominal" steps. However, this approach would be rather penalizing for the Metropolis algorithm since it would take, on average, $O(N \log N)$ steps to give all vertices of the graphs at least one opportunity to be selected. Another opportunity would be to count the number of "attempted flips". Since in both $\mathcal{A}$ and $\mathcal{S}$ all components are potentially updated, following this approach would require setting the number of steps for both $\mathcal{A}$ and $\mathcal{S}$ to be the same of the number of sweeps for $\mathcal{M}$. However, if run on a single core, the computing time for a step of $\mathcal{A}$ of $\mathcal{S}$ is significantly shorter than the time required for a sweep of $\mathcal{M}$ (see Table 1). Finally one could simply consider computation time. However, we feel that this criterion would be too dependent on the implementation details and the features of the computing machine. Because of these considerations, we decided to use a "hybrid" approach using as a primary criterion the total number of attempted flips but giving an extra quota of iterations to the parallel dynamics (note that, in each full step, the Shaken dynamics tries to update all component twice) to take into account (in a conservative way) their faster execution times.

 doi:10.20944/preprints202408.2259.v1

Table 1. This table shows the speedups with respect to Metropolis algorithm of the $\mathcal{A}$ and $\mathcal{S}$ algorithms. The speedup is computed taking into account the execution time of 1000 iterations of each algorithm (sweeps for $\mathcal{M}$ and steps for $\mathcal{A}$ and $\mathcal{S}$. Computation times are determined by computing averages over 5 runs. For each of the two parallel algorithms, columns with headings 1, 2, 4, and 8 give the speedup when the number of threads used by the BLAS library is set to 1, 2, 4, and 8 respectively. Column N gives the number of nodes in the graph.

graph id	N	$\mathcal{A}$				$\mathcal{S}$			
		1	2	4	8	1	2	4	8
C1000.9	1000	10.2	18.3	127.0	177.2	3.1	5.6	46.7	50.5
C125.9	125	8.2	8.0	8.0	8.0	3.3	3.7	4.0	3.4
C2000.5	2000	7.7	14.4	25.8	48.6	2.2	4.1	7.6	12.2
C2000.9	2000	5.7	7.0	12.0	19.6	1.6	2.0	3.6	6.1
C250.9	250	22.3	24.5	16.5	18.3	9.0	8.6	6.9	7.9
C4000.5	4000	5.5	6.8	12.6	22.3	1.5	1.9	3.6	6.0
C500.9	500	37.9	60.3	84.8	107.3	14.3	21.7	31.0	42.3
DSJC1000_5	1000	6.0	10.7	85.0	128.9	1.8	3.3	6.3	44.8
DSJC500_5	500	30.9	47.1	66.4	82.9	11.2	16.7	24.3	32.6
MANN_a27	378	28.7	41.1	55.6	66.4	10.6	15.6	21.5	26.4
MANN_a45	1035	6.1	10.8	87.4	126.6	1.8	3.3	6.2	23.7
MANN_a81	3321	6.3	12.0	22.3	40.3	2.5	3.4	6.6	12.4
brock200_2	200	29.0	27.0	33.3	36.1	11.7	13.8	13.5	14.5
brock200_4	200	29.9	27.4	33.3	36.3	12.4	10.5	13.0	13.9
brock400_2	400	33.7	31.8	43.1	56.2	12.5	17.1	16.6	21.0
brock400_4	400	33.0	43.2	42.8	54.4	12.4	17.0	15.8	20.9
brock800_2	800	38.1	59.7	57.0	79.8	12.1	20.0	20.3	29.4
brock800_4	800	37.5	59.4	54.8	79.6	1.8	19.6	20.3	29.4
gen200_p0.9_44	200	27.6	25.8	31.9	34.4	11.5	13.8	13.0	14.5
gen200_p0.9_55	200	29.5	35.6	32.0	34.2	11.8	14.1	13.3	14.4
gen400_p0.9_55	400	33.6	31.5	38.0	48.8	13.5	12.4	16.1	22.4
gen400_p0.9_65	400	35.7	30.2	49.0	58.3	12.9	14.6	14.9	21.7
gen400_p0.9_75	400	34.0	46.0	45.6	58.1	13.3	16.0	17.3	22.4
hamming10-4	1024	5.8	66.1	66.6	94.5	1.8	3.5	21.4	32.7
hamming8-4	256	25.0	28.8	32.1	35.4	7.6	10.8	12.5	15.0
keller4	171	22.6	23.3	23.2	23.2	9.0	10.6	10.3	11.3
keller5	776	39.1	59.4	58.9	84.3	13.1	15.8	19.5	28.5
keller6	3361	5.5	6.8	12.6	23.0	1.4	2.6	3.5	6.6
p_hat1500-1	1500	5.7	10.3	102.4	94.4	1.7	3.1	5.6	6.2
p_hat1500-2	1500	5.8	10.7	98.0	108.4	2.7	18.5	35.4	41.9
p_hat1500-3	1500	15.3	26.7	38.1	49.7	4.6	7.5	11.4	14.5
p_hat300-1	300	13.4	12.7	17.5	8.3	5.4	6.2	7.8	8.7
p_hat300-2	300	14.2	20.6	25.8	17.6	4.6	7.6	7.2	8.8
p_hat300-3	300	12.1	12.9	16.3	18.2	5.0	5.2	6.7	7.9
p_hat700-1	700	15.6	22.8	28.4	31.1	5.1	7.2	9.4	11.1
p_hat700-2	700	15.1	23.3	30.6	28.0	5.2	6.0	9.7	11.4
p_hat700-3	700	16.9	22.6	28.8	32.9	5.7	7.9	8.3	11.8

The three algorithms were implemented in Julia (version 1.10, see [21]) and executed on an Nvidia DGX-1 system equipped with Intel Xeon CPU E5-2698 v4 @ 2.20GHz (80 cores). We considered only a CPU implementation of the algorithms. For linear algebra operations (matrix-vector products) we used BLAS libraries ([22,23]) with the default implementation shipped with the LinearAlgegra package of Julia. Random numbers were generated using the default random number generator available in Julia (which in version 1.10 implements the Xoshiro256++ algorithm [24]).

 doi:10.20944/preprints202408.2259.v1

Table 2. This table shows the largest clique found with each algorithm for the examined benchmark graphs. Bold numbers highlight those cases where the cardinality of the largest clique found by any of the algorithms considered in this paper coincides with the clique number of the graph or its best lower bound available in the literature (to the best of our knowledge). Underlined numbers signal the largest clique obtained with any of the algorithms considered in the paper when this number is smaller than the clique number of the graph (or its best lower bound). Column $\mathcal{A}$ gives the results for the Probabilistic Cellular Automaton, column $\mathcal{S}$ the results for the Shaken dynamics, and column $\mathcal{M}$ the results of the Metropolis single flip dynamics. Column $\omega(G)$ gives the value of the clique number of the graph (or its best lower bound). The value of $\omega(G)$ is marked with an asterisk when it is known to be exact (see [2]). Column N gives the number of nodes in the graph.

graph id	N	$\mathcal{A}$	$\mathcal{S}$	$\mathcal{M}$	$\omega(G)$
C1000.9	1000	67	65	**68**	68
C125.9	125	**34**	**34**	**34**	34
C2000.5	2000	**16**	15	**16**	16*
C2000.9	2000	73	72	76	80
C250.9	250	**44**	**44**	**44**	44
C4000.5	4000	17	16	17	18*
C500.9	500	**57**	**57**	**57**	57
DSJC1000_5	1000	**15**	14	**15**	15*
DSJC500_5	500	**13**	**13**	**13**	13*
MANN_a27	378	124	123	124	126*
MANN_a45	1035	335	334	336	345*
MANN_a81	3321	1084	1080	1083	1100*
brock200_2	200	**12**	**12**	**12**	12*
brock200_4	200	**17**	**17**	**17**	17*
brock400_2	400	**29**	25	**29**	29*
brock400_4	400	**33**	32	**33**	33*
brock800_2	800	20	20	21	24*
brock800_4	800	20	20	21	26*
gen200_p0.9_44	200	**44**	**44**	**44**	44
gen200_p0.9_55	200	**55**	**55**	**55**	55
gen400_p0.9_55	400	**55**	**55**	**55**	55
gen400_p0.9_65	400	**65**	**65**	**65**	65
gen400_p0.9_75	400	**75**	**75**	**75**	75
hamming10-4	1024	**40**	**40**	**40**	40
hamming8-4	256	**16**	**16**	**16**	16
keller4	171	**11**	**11**	**11**	11*
keller5	776	**27**	**27**	**27**	27*
keller6	3361	55	55	**59**	59*
p_hat1500-1	1500	**12**	11	**12**	12*
p_hat1500-2	1500	**65**	**65**	**65**	65*
p_hat1500-3	1500	**94**	**94**	**94**	94*
p_hat300-1	300	**8**	**8**	**8**	8*
p_hat300-2	300	**25**	**25**	**25**	25*
p_hat300-3	300	**36**	**36**	**36**	36*
p_hat700-1	700	**11**	**11**	**11**	11*
p_hat700-2	700	**44**	**44**	**44**	44*
p_hat700-3	700	**62**	**62**	**62**	62*

The Metropolis algorithm finds the best-known clique 30 out of 37 times proving to be, in practice, a rather effective tool for the maximum clique problem for instances with a size that is not too large. The PCA algorithm $\mathcal{A}$ allowed to find the maximum clique 27 out of 37 times whereas the Shaken dynamics found the maximum clique 23 out of 37 times. In those cases where neither of the algorithms found the maximum clique the Metropolis is the one which found the largest clique more often. It should be noted, though, that in the case of the graph `MANN_a81`, the PCA found a better solution of the Metropolis and in two more cases it found an equivalent solution.

 doi:10.20944/preprints202408.2259.v1

In general, the performances of the two parallel algorithms, especially those of the PCA $\mathcal{A}$, are not much worse in terms of the value of the largest clique found. Nevertheless, it is possible to argue that the slightly worse performances of $\mathcal{A}$ and $\mathcal{S}$ are partly compensated by their computational effectiveness. Even using a single thread for the computation of the energy field, algorithm $\mathcal{A}$ was faster between 5 and 40 times. Running with 8 threads the speedup reached a factor beyond 100 for certain instances and it was more than 50 the majority of times. The situation is not too different for the Shaken dynamics $\mathcal{S}$. Here one has to take into account that, at each step, two configurations are visited: one for each half-step. A GPU implementation of the two algorithms is expected to yield an even greater speed up, especially on the larger instances. The increased speed could be advantageous in applications.

Looking at the two parallel algorithms, $\mathcal{A}$ appears to be better than $\mathcal{S}$ both in terms of the value of the solution provided and in terms of speed (even though the Shaken dynamics explores two configurations at each step the time to get a new configuration with the PCA appears to be less than $\frac{1}{2}$ of the time required by the shaken dynamics). Likely this can be explained by the fact that the energy fields of the PCA can be computed as the $\tilde{J} \cdot \eta$ matrix-vector product where the matrix $\tilde{J}$ is symmetric, whereas the matrix $\overleftarrow{J}$ used to compute $\overleftarrow{J} \cdot \eta$ is not.

4. Discussion

We believe that the results presented above foster a further investigation into the behavior of Probabilistic Cellular Automata. Indeed, they appear to provide good results while taking great advantage of the features of the computing architectures available nowadays. This investigation should start, in our opinion, from a better understanding of the relation between the stationary measure of the PCA and the Gibbs measure. In particular, it would be interesting to determine, for any $\bar{\beta}$, bounds for the parameters (β, q) so that the total variation distance between $\pi_{\mathcal{A}}$ (or $\pi_{\mathcal{S}}$) and the Gibbs measure at inverse temperature $\bar{\beta}$ is within a prescribed value ε.

In the previous section, we observed that the values of the largest clique obtained with the Metropolis algorithm $\mathcal{M}$ are slightly better than those obtained with $\mathcal{A}$ and $\mathcal{S}$. It should be mentioned that the dependence on two parameters of the latter algorithms makes their tuning more challenging than the tuning of the single parameter of the Metropolis. For this reason, it would be interesting to study, from a theoretical point of view, the behavior of both dynamics in the (β, q) plane. In particular, it would be interesting to determine a "phase transition" curve identifying a "high temperature" region where the stationary measure is "close" to the uniform measure on $\mathcal{X}$ and a "low temperature" where the stationary measure is "concentrated" on the minima of the Hamiltonian 1. It is to be expected that this phase transition curve will depend on the law with which the graph has been generated An analog study would be interesting for the Metropolis case where it would be interesting to determine a critical value of β separating the high and low-temperature regions.

As already mentioned, the Metropolis algorithm discussed above is equivalent, at large inverse temperature, to the algorithm described by Jerrum in [4]. In that paper, it has been shown that the mixing time of the chain, in the case of Erdős-Rényi graphs, is not polynomial in the number of vertices. Further, the limiting typical size of clique typically determined by the Metropolis process has been determined. It would be interesting to perform a similar analysis for the two parallel algorithms introduced here.

From the computational point of view, the most expensive task of the PCA and the Shaken dynamics is the computation of the vector of fields which is of the type $h(\eta) = J \cdot \eta$. This matrix-vector product can be performed using highly optimized linear algebras libraries, which may take great advantage of multicore processors and GPUs.

As a consequence, the algorithm will benefit *for free* from improvements in the performances of linear algebra libraries. Since the use of these libraries is ubiquitous in computer applications, improvements in this direction will be driven by the efforts of communities coming from the most diverse domains.

 doi:10.20944/preprints202408.2259.v1

Note this type of advantage could be exploited even further. Indeed, it is possible to let k PCA evolve in parallel. Denoting by $\vec{E}$ the matrix whose columns are the configurations of the k chains, the collection of the k vector of *fields* $\{h_i(\eta^{(m)})\}_{i=1,\dots,N;m=1,\dots,k}$ can be computed as the matrix matrix product $\tilde{J} \cdot \vec{E}$. Performing this operation is, generally, significantly faster than the computation of k matrix-vector products. Note that, for each chain, a different pair of parameters β and q could be used without significantly affecting the computation time.

The possibility to run multiple chains at the time with different inverse temperatures β makes it possible to consider both dynamics $\mathcal{A}$ and $\mathcal{S}$ for algorithms like parallel tempering [25] which has been applied to the clique problem using standard single spin Metropolis dynamics (see, e.g. [7]).

Note that Markov Chain Monte Carlo algorithms defined in terms of a "pair Hamiltonian" were already considered in [5,26,27]. Also, the algorithm considered in those works (referred to as "Cavity") had the feature of updating more than one component of the configuration at each step. However, in these works, the cardinality of the configuration stayed fixed throughout the whole evolution of the chain and, therefore, its components could not be updated independently limiting, in this way, the possibility to exploit parallel computing architectures at their fullest.

As a final comment, we would like to mention that both the parallel algorithms discussed in this paper may have applications outside combinatorial optimization. As far as the PCA is concerned, it has been suggested in [28] that this Markov Chain could be used to sample Exponential Random Graphs in the generation of Synthetic Power Grids. For what concern the shaken dynamics, it has been hinted in [17,29] has a microscopic for tidal dissipation.

Funding: This work has been partially supported by PRIN 2022 PNRR: "RETINA: REmote sensing daTa INversion with multivariate functional modeling for essential climAte variables characterization" (Project Number: P20229SH29, CUP: J53D23015950001) funded by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, under the Italian Ministry of University and Research (MUR).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: We downloaded the instances of the graphs analyzed in this paper from https://iridia.ulb.ac.be/~fmascia/maximum_clique/ Information on the generators used to create the instances and their source code can be found at DIMACS ftp reachable at http://archive.dimacs.rutgers.edu/pub/challenge/graph/.

Acknowledgments: The author thanks the Department of Mathematics and Physics of the University of Rome "Tre" which provided computing resources.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MCMC	Monte Carlo Markov Chain
PCA	Probabilistic cellular automaton or Probabilistic cellular automata
PCAMC	PCA Markov Chain
QUBO	Quadratic Unconstrained Binary Optimization

References

1. Karp, R.M., Reducibility among Combinatorial Problems; Springer US: Boston, MA, 1972; pp. 85–103.
2. Marino, R.; Buffoni, L.; Zavalnij, B. A Short Review on Novel Approaches for Maximum Clique Problem: from Classical algorithms to Graph Neural Networks and Quantum algorithms. *arXiv preprint arXiv:2403.09742* **2024**.
3. Häggström, O. *Finite Markov chains and algorithmic applications*; Vol. 52, Cambridge University Press, 2002.
4. Jerrum, M. Large cliques elude the Metropolis process. *Random Structures & Algorithms* **1992**, *3*, 347–359.
5. Iovanella, A.; Scoppola, B.; Scoppola, E. Some spin glass ideas applied to the clique problem. *Journal of Statistical Physics* **2007**, *126*, 895–915.
6. Montanari, A. Finding one community in a sparse graph. *Journal of Statistical Physics* **2015**, *161*, 273–299.

7. Angelini, M.C. Parallel tempering for the planted clique problem. *Journal of Statistical Mechanics: Theory and Experiment* **2018**, *2018*, 073404.
8. Punnen, A.P. The quadratic unconstrained binary optimization problem. *Springer International Publishing* **2022**, *10*, 978–3.
9. Glover, F.; Kochenberger, G.; Du, Y. Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models. *Annals of Operations Research* **2019**, *314*, 141–183.
10. Baioletti, M.; Santini, F. Abstract Argumentation Goes Quantum: An Encoding to QUBO Problems. Pacific Rim International Conference on Artificial Intelligence. Springer, 2022, pp. 46–60.
11. Glover, F.; Kochenberger, G.; Ma, M.; Du, Y. Quantum Bridge Analytics II: QUBO-Plus, network optimization and combinatorial chaining for asset exchange. *Annals of Operations Research* **2022**, *314*, 185–212.
12. Tasseff, B.; Albash, T.; Morrell, Z.; Vuffray, M.; Lokhov, A.Y.; Misra, S.; Coffrin, C. On the emerging potential of quantum annealing hardware for combinatorial optimization. *Journal of Heuristics* **2024**, pp. 1–34.
13. Fukushima-Kimura, B.H.; Handa, S.; Kamakura, K.; Kamijima, Y.; Kawamura, K.; Sakai, A. Mixing time and simulated annealing for the stochastic cellular automata. *Journal of Statistical Physics* **2023**, *190*, 79.
14. Scoppola, B.; Troiani, A. Gaussian Mean Field Lattice Gas. *Journal of Statistical Physics* **2018**, *170*, 1161–1176.
15. Isopi, M.; Scoppola, B.; Troiani, A. On some features of Quadratic Unconstrained Binary Optimization with random coefficients. *arXiv preprint arXiv:2402.17059* **2024**.
16. Apollonio, V.; D'autilia, R.; Scoppola, B.; Scoppola, E.; Troiani, A. Criticality of Measures on 2-d Ising Configurations: From Square to Hexagonal Graphs. *Journal of Statistical Physics* **2019**, *177*, 1009–1021.
17. Apollonio, V.; D'Autilia, R.; Scoppola, B.; Scoppola, E.; Troiani, A. Shaken dynamics: an easy way to parallel Markov Chain Monte Carlo. *Journal of Statistical Physics* **2022**, *189*, 39.
18. D'Autilia, R.; Andrianaivo, L.N.; Troiani, A. Parallel simulation of two-dimensional Ising models using probabilistic cellular automata. *Journal of Statistical Physics* **2021**, *184*, 1–22.
19. Scoppola, B.; Troiani, A.; Veglianti, M. Shaken dynamics on the 3d cubic lattice. *Electronic Journal of Probability* **2022**, *27*, 1–26.
20. Johnson, D.S.; Trick, M.A. *Cliques, coloring, and satisfiability: second DIMACS implementation challenge, October 11-13, 1993*; Vol. 26, American Mathematical Soc., 1996.
21. Bezanson, J.; Edelman, A.; Karpinski, S.; Shah, V.B. Julia: A fresh approach to numerical computing. *SIAM review* **2017**, *59*, 65–98.
22. Lawson, C.L.; Hanson, R.J.; Kincaid, D.R.; Krogh, F.T. Basic linear algebra subprograms for Fortran usage. *ACM Transactions on Mathematical Software (TOMS)* **1979**, *5*, 308–323.
23. Dongarra, J.J.; Croz, J.D.; Hammarling, S.; Hanson, R.J. An extended set of FORTRAN basic linear algebra subprograms. *ACM Trans. Math. Softw.* **1988**, *14*, 1–17.
24. Blackman, D.; Vigna, S. Scrambled linear pseudorandom number generators. *ACM Transactions on Mathematical Software (TOMS)* **2021**, *47*, 1–32.
25. Hukushima, K.; Nemoto, K. Exchange Monte Carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan* **1996**, *65*, 1604–1608.
26. Viale, M. Il problema della massima clique: teoria & pratica. *PhD thesis* **2009**.
27. Gaudilliere, A.; Scoppola, B.; Scoppola, E.; Viale, M. Phase transitions for the cavity approach to the clique problem on random graphs. *Journal of statistical physics* **2011**, *145*, 1127–1155.
28. Giacomarra, F.; Bet, G.; Zocca, A. Generating Synthetic Power Grids Using Exponential Random Graph Models. *PRX Energy* **2024**, *3*, 023005.
29. Pinzari, G.; Scoppola, B.; Veglianti, M. Spin orbit resonance cascade via core shell model. Application to Mercury and Ganymede. *arXiv preprint arXiv:2402.07650* **2024**.