Approximating Maximum Clique with a Hopfield Network

Arun Jagota
Department of Computer Science,
State University Of New York At Buffalo, USA
e-mail : jagota@cs.buffalo.edu

Technical Report 92-33

Abstract

In a graph, a *clique* is a set of vertices such that every pair is connected by an edge. MAX-CLIQUE is the optimization problem of finding the largest clique in a given graph, and is *NP*-hard, even to approximate well. Several real-world and theory problems can be modeled as MAX-CLIQUE. In this paper, we efficiently approximate MAX-CLIQUE in a special case of the Hopfield Network whose stable states are maximal cliques. We present several energy-descent optimizing dynamics; both discrete (deterministic and stochastic) and continuous. One of these emulates, as special cases, two well known greedy algorithms for approximating MAX-CLIQUE. We report on detailed empirical comparisons on random graphs. Mean-Field Annealing—an efficient approximation to Simulated Annealing—and a stochastic dynamics are the narrow but clear winners. All dynamics approximate much better than one which emulates a "naive" greedy heuristic.

1 Cliques and Maximum Clique

In a graph with undirected edges, a *clique* is a set of vertices such that every pair is connected by an edge. A clique is **maximal** if no strict superset of it is also a clique. A k-clique is a clique of size k. A clique is **maximum** if it is the largest clique. In Figure 1, the vertex sets: {a,c,d,e}, {c,d},{a,b}, and {c,d,e} are not clique, non-maximal 2-clique, maximal-but-not-maximum clique, and maximum clique (size 3) respectively.

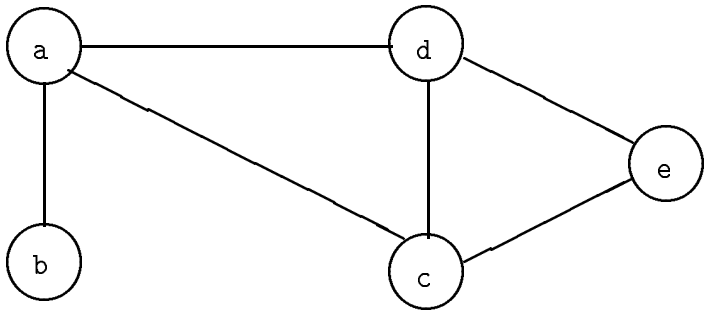

Figure 1. Graph with maximal cliques {a,b}, {a,c,d}, and {c,d,e}

MAX-CLIQUE is the optimization problem of finding a largest clique in a given graph and is *NP*-hard [26] even to approximate [1] (also see [10, 9, 7]). That is, unless *P=NP*, which is highly unlikely, there is no algorithm that, for *every* given graph, can find a maximum clique—or even a clique within, for example, a constant factor of it—tractably (i.e. in time polynomial in the size of the graph).

Several problems can be efficiently reduced to MAX-CLIQUE, and thus solved by algorithms for MAX-CLIQUE. One example is satisfiability of Boolean Formulae (see [10, 7]). Many other problems are in principle efficiently reducible to MAX-CLIQUE because of its *NP*-hardness. Some—but not all—reductions also preserve approximability. In such cases, reduced problems can also be approximated, by approximating MAX-CLIQUE; this being relatively easier. The optimization problem of finding the largest number of simultaneously satisfiable clauses is one such problem [7].

Several real-world problems can also be modeled as exactly-solving or approximating MAX-CLIQUE. These include Constraint Satisfactions Problems [22] (useful in AI; see [31]), object recognition modeled as subgraph isomorphism; experimental design; and signal transmission [3], information retrieval [2], clustering in pattern recognition [8], labeled point pattern matching [32], stereo vision correspondence [17], and PLA folding [29].

The topic of this paper is to approximately solve MAX-CLIQUE instances in a Hopfield network special case. The motivating factors for this work are:

1. MAX-CLIQUE is of theoretical and practical significance, as discussed above.

2. The Hopfield network special case that we employ has been used earlier on associative memories [18, 21]. The MAX-CLIQUE problem arises naturally on it. By applying the same network to both problems, common characteristics can be studied. For example, steepest descent dynamics turns out to be useful on both problems.

3. The problem is easy to map to neural networks (as compared with TSP). In particular, our energy function does not admit any invalid solutions.

We are aware of the following earlier works on Maximum Clique using neural networks. The problem appears to have been first encoded in Hopfield networks in [4, 11]. The weights are binary and no invalid solutions are admitted. In [4], experimental results are reported on Maximum Independent Set—the complement problem—but the focus is on running time rather than solution quality. A different encoding is in [35]. No experimental results are reported. Their encoding admits invalid solutions and employs non binary weights. The Maximum Independent Set problem is indirectly formulated in [37], with essentially the same encoding as in [11]. No experiments are reported. The approach of the current paper was presented in [20] and further studied in [23]. The weights are binary and no invalid solutions are admitted. Recently, another encoding of Maximum Clique is in [30]. The weights are non binary. Good experimental performance is reported on random graphs. Our Mean Field Annealing and stochastic dynamics performances (Table I) are competitive if not slightly better. A stronger comparison requires that (a) the same test graphs are used and (b) "hard" graphs are included, as we have done, in the test set (because MAX-CLIQUE is somewhat easy to approximate on random graphs).

2 Hopfield-clique Network

The Hopfield Network [15, 16] is a recurrent N-unit network closely related to Ising Spin Models. Units are connected with symmetric real-valued weights w_{ij}. In the discrete Hopfield Network [15], unit states are: $S_i \in \{0, 1\}$ (a different formulation has $\{-1, 1\}$). Unit i receives external input (bias) I_i. The input to unit S_i is:

$$n_i := \sum_j w_{ij} S_j + I_i \tag{1}$$

Any serial-update (asynchronous) rule of the form: At time $t + 1$, pick one unit and update its state as follows:

$$S_i(t+1) := \begin{cases} 1 & \text{if } n_i(t) > 0 \\ 0 & \text{if } n_i(t) < 0 \\ S_i(t) & \text{otherwise} \end{cases} \tag{2}$$

minimizes the following energy function whenever a unit switches:

$$E = -\frac{1}{2} \sum_{ij} w_{ij} S_i S_j - \sum_i I_i S_i \tag{3}$$

In the continuous Hopfield Network [16], units are continuous: $S_i \in [0, 1]$ (a different formulation has $[-1, 1]$). S_i are updated via a continuous nonlinear dynamics, described later, which minimizes the following energy function [16]:

$$E = -\frac{1}{2} \sum_{ij} w_{ij} S_i S_j - \sum_i I_i S_i + \sum_i \int_0^{S_i} g_\lambda^{-1}(S) dS \tag{4}$$

where $g_\lambda(x) \equiv \frac{1}{1 + e^{-\lambda x}}$ is a sigmoid and λ its gain. Convergence to a continuous local minimum of (4) is thus guaranteed. Hopfield [16] also related the continuous local minima of (4) to the discrete local minima of (3). He observed that the continuous and discrete local minima of the first term of (4) are the same and showed that the contribution of the second term fades as λ is increased.

That is, as λ is increased, the continuous local minima of (4) monotonically approach the discrete local minima of (3). This result will be significant in this paper.

The (binary-weights) Hopfield-clique Network (HcN) [19, 21] is a *fully-connected* special case of the Hopfield Network with the following restriction on weights: $w_{ij} \in \{\rho, 1\}$, $i \neq j$ and $\rho < 0$. There are no self-weights ($w_{ii} = 0$). $I_i = w_0$ is the same external bias to all units. $G_N = (V, E)$ is the graph underlying the weight matrix whose vertices are the units and there is an undirected edge between every pair of units with positive weights ($w_{ij} = w_{ji} = 1$) and no edges for the remaining pairs.

The discrete HcN has $S_i \in \{0, 1\}$. The network state vector $S = (S_i) \in \{0, 1\}^N$ is the characteristic membership vector of some $V' \subseteq V$. We use S to denote the network state as a set as well as its characteristic state vector. S is a locally minimum (stable) state of the energy function of (3) if the energy of every state neighboring S (Hamming distance of 1 from S) is greater than or equal to that of S.

Lemma 1 ([19, 21]) *For $\rho < -N$ and $0 < w_0 < 1$, the stable states of discrete HcN are exactly the maximal cliques of the G_N underlying it.*

We defer the proof to Section 3.2 where it follows as a corollary of Lemma 3.

By Lemma 1, the stable states of the discrete HcN instance with the underlying graph of Figure 1 are: {a,b}, {a,c,d}, and {c,d,e}, the maximal cliques. The third stable state is: units c,d,e are ON; the rest OFF. Lemma 1 is the basis for applying the discrete HcN to MAX-CLIQUE.

A network with 0/1 weights has stable states equivalent to HcN [13] and has been implemented in optics [36].

The continuous HcN has $S_i \in [0, 1]$. Combining the Hopfield result relating continuous and discrete minima with Lemma 1 gives: "As λ is increased, the stable states of a continuous HcN monotonically approach (characteristic vectors of) maximal cliques of its underlying G_N". This is the basis for applying continuous HcN to MAX-CLIQUE.

The following storage rule, developed earlier and used for associative memories (see [19, 18]), can be used to make any given graph G the graph underlying HcN. We present this rule because it is on-line, may have efficient implementations, and is a single mechanism for storing associative memories and graphs. The initial weight state is: For all $i \neq j$, $w_{ij}(0) = \rho$. Any set of vertices $V' \subseteq V$ is stored in HcN at time t as follows. For all $i \neq j$,

$$w_{ij}(t+1) \;:=\; \begin{cases} 1 & \text{if } S_i \text{ and } S_j \in V' \\ w_{ij}(t) & \text{otherwise} \end{cases} \tag{5}$$

3 Discrete Descent Dynamics

Lemma 2 *[21] For $\rho < -2N$, with any discrete serial-update dynamics, HcN converges from any initial state in $\leq 2N$ unit-switches.*

Proof-sketch: The main idea in the proof is that if a unit switches ON, it cannot subsequently switch OFF. Hence no unit switches more than twice. The following description of this main idea's proof, shorter than the rigorous one in [21], is due to T. Grossman [13]. Suppose unit i switches ON at time t. All units j such that $w_{ij} = \rho$ are OFF at time t. At time $t' > t$, because i is ON, none of these units can switch ON. Hence i cannot switch OFF at time $t' + 1$ $\Box$.
A result equivalent to Lemma 2 was also independently obtained in [37].

3.1 Steepest Descent

One discrete dynamics that we will use to approximate MAX-CLIQUE is serial-update steepest energy-descent (SD). In every cycle, exactly one unit i—the one whose switch would maximally decrease the energy—is first picked and then switched. Specifically, i satisfies:

$$\Delta E_i = \min_j \Delta E_j < 0$$

where

$$\Delta E_k(t) = -[S_k(t) - S_k(t-1)]n_k$$

is the energy change caused by the switch (i.e. $S_k(t) \neq S_k(t-1)$) of S_k.

With $\rho < -2N$, SD emulates the following greedy clique-finding algorithm on the underlying graph G_N [20]:

---

1. $S \leftarrow S_0$;
2. <u>while</u> S is not a clique of G_N <u>do</u>
   (a) Pick a $v \in S$ with minimum degree in $G[S]$ (i.e. $d_{G[S]}v = \delta(G[S])$)
   (b) $S \leftarrow S \setminus \{v\}$;
3. <u>end</u>;
4. <u>while</u> S is not a maximal clique of G_N <u>do</u>
   (a) Pick $v \in V \setminus S$ such that $S \cup \{v\}$ is a clique of G_N;
   (b) $S \leftarrow S \cup \{v\}$;
5. <u>end</u>;

SD($S_0, \rho < -2N$)

From any initial state S_0, this algorithm does all the switch OFF's before any switch ON's. Step 2a switches a unit OFF that is inhibited by the *largest* number of ON units. Step 4a switches a unit ON if it is currently OFF and receives no inhibition from any of the ON units. With $S_0 \leftarrow V$ (initial state: all units ON), this becomes a well known greedy algorithm for finding a large clique (see [12]). With $S_0 \leftarrow \emptyset$ (initial state: all units OFF), this becomes another well known "naive" greedy algorithm for finding a large clique (see [27]). Though it follows from Lemma 2, it is also clear from the algorithm that $SD(S_0)$ from any initial state S_0 converges in $\leq 2N$ unit-switches.

We sketch the proof that SD emulates the above algorithm. Consider a state S that is not a clique. First, because ρ is sufficiently negative and because of steepest descent, a unit in S necessarily switches OFF even if there is a unit in $V \backslash S$ that is adjacent to every vertex in S (the latter would also decrease energy). Second, because of steepest descent, the switched OFF unit must have minimum degree in $G[S]$. Consider a state S that is a clique. Because ρ is sufficiently negative, only a unit in $V \backslash S$ that is adjacent to every unit in S may be switched ON $\Box$.

Example. For the HcN of Figure 1, with $S_0 = \{a, b, c, d, e\}$, the algorithm dynamics is:

$$\{a, b, c, d, e\} \overset{Steps\,2a,b}{\longrightarrow} \{a, c, d, e\} \overset{Steps\,2a,b}{\longrightarrow} \{c, d, e\}$$

The second transition could also be to {a,c,d}.

3.2 ρ-annealing

The second discrete dynamics that we will use on MAX-CLIQUE is applicable only to HcN and is based on annealing (varying) ρ while performing SD. We term it ρ-annealing. In ρ-annealing we (1) start from the initial state: all vertices ON, with a negative value of ρ very close to 0, (2) reach a stable state by SD, and (3) decrease ρ. We repeat Steps 2,3—using the stable state of the previous Step 2 as the initial state of the next Step 2—until $\rho < -N$ when, by Lemma 1, the stable state is a maximal clique.

---

1. $\rho \leftarrow -\epsilon \qquad [= -\frac{1}{8}]$;
2. $S \leftarrow SD(V, \rho)$;
3. <u>while</u> $\rho > -N$ <u>do</u>
   (a) Decrease $\rho \qquad [\rho \leftarrow 2\rho]$;
   (b) $S \leftarrow SD(S, \rho)$;
4. <u>end</u>

ρ-annealing

At every ρ, convergence is guaranteed by the Hopfield convergence theorem [15]. The square brackets indicate the choices that we used in our experiments (see Section 5).

Analysis of ρ-annealing. V is the set of vertices (units). Let $d(v)|S \equiv$ number of vertices of $S \subseteq V$ that v is adjacent (has an edge) to. Let $\overline{d}(v)|S \equiv$ number of vertices of $S \subseteq V$ that v is not adjacent to. In the graph of Figure 1, $d(c)|\{a,b,c,d\} = 2$; $\overline{d}(e)|\{a,b,c,d\} = 2$.

When the network state is S (the set of ON units), the input n_v to a unit (vertex) v is (rewriting (1) for HcN):

$$n_v(S) := \rho\overline{d}(v)|S + d(v)|S + w_0 \tag{6}$$

The role of w_0 is to ensure that if $\rho\overline{d}(v)|S + d(v)|S = 0$ then $n_v(S) > 0$ while preserving $sgn(n_v(S)) = sgn(\rho\overline{d}(v)|S + d(v)|S + w_0)$ where $sgn(x) = 1$ if $x \geq 0$ and $sgn(x) = -1$ otherwise. This ensures that a unit switches in every cycle (cf. (2)). Without loss of generality, we restrict ourselves to rational $\rho = -\frac{x}{y}$ where x and y are positive integers. By noting that

$$n_v(S) = \frac{-x\overline{d}(v)|S + yd(v)|S}{y} + w_0$$

it is easy to see that choosing $w_0 : 0 < w_0 < \frac{1}{y}$ satisfies the constraints.

Definition 1 *$S \subseteq V$ is a k-degree-ratio set if: (a) $\forall v \in S : \frac{d(v)|S}{\overline{d}(v)|S} \geq k$ and (b) $\forall v \in V \backslash S : \frac{d(v)|S}{\overline{d}(v)|S} < k$.*

In Figure 1, {a,c,d,e} is a 2-degree-ratio set but not a 3-degree-ratio set.

Lemma 3 *The HcN stable states for arbitrary $\rho < 0$ and $0 < w_0 < \frac{1}{y}$ are exactly the $|\rho|$-degree-ratio sets of G_N.*

Proof: From (6) and Definition 1, it follows that in state S: An ON unit $v \in S$ can switch OFF iff $\frac{d(v)|S}{\overline{d}(v)|S} < |\rho|$; An OFF unit $v \in V \backslash S$ can switch ON iff $\frac{d(v)|S}{\overline{d}(v)|S} \geq |\rho|$; Hence no unit can switch iff S is a $|\rho|$-degree-ratio set $\Box$.

With $\rho = -2$, the stable states of the HcN with the underlying graph of Figure 1 are: {a,c,d,e} and {a,b}, the 2-degree-ratio sets.

Lemma 1—for $\rho < -N$ and $0 < w_0 < 1$, the HcN stable states are exactly the maximal cliques of G_N—follows as a corollary because $y = 1$ and k-degree-ratio sets for any $k > N$ are maximal cliques by definition.

We can now discuss the expected behavior of ρ-annealing. At every ρ, the stable states are $|\rho|$-degree-ratio sets. At ρ close to 0, they are expected to satisfy constraints only weakly (i.e. not be cliques). As ρ is decreased, constraints will become tighter until the stable states eventually become maximal cliques. The idea of annealing ρ is motivated by the hope that—in analogy with simulated annealing—a progression of stable states with monotonically tightening constraints will eventually lead to a large clique. For the HcN example of Figure 1, the sequence of stable states during the algorithm run is:

$$\{a,b,c,d,e\} \stackrel{\rho=-0.125}{\longrightarrow} \{a,b,c,d,e\} \stackrel{\rho=-0.25}{\longrightarrow} \{a,b,c,d,e\} \stackrel{\rho=-0.5}{\longrightarrow} \{a,c,d,e\} \stackrel{\rho=-1}{\longrightarrow} \{a,c,d,e\} \stackrel{\rho=-2}{\longrightarrow} \{a,c,d,e\} \stackrel{\rho=-4}{\longrightarrow} \{c,d,e$$

where the last stable state could also be {a,c,d}.

Though the above analysis may suggest that ρ-annealing and $SD(V, < -2N)$ are quite different heuristics, our empirical comparisons however led to the surprising observation that, on the same graph, they both exhibited identical behavior (switch decisions) upto a certain point. This led to the following result. Ties are broken lexicographically in both ρ-annealing and $SD(V, \rho)$.

Lemma 4 *If $SD(S, \rho)$ switches a unit OFF from state S then, for any $\rho' < \rho$, $SD(S, \rho')$ also switches the same unit OFF.*

Proof: Let v be the switched OFF unit in $SD(S, \rho)$. Consider $u \in S$ such that $n_u(S) < 0$.

$$\rho\overline{d}(v)|S+d(v)|S+w_0 \stackrel{n_v(S) \leq n_u(S) < 0}{\leq} \rho\bar{d}(u)|S+d(u)|S+w_0 \stackrel{\rho' \leq \rho < 0}{\Longrightarrow} \rho'\overline{d}(v)|S+d(v)|S+w_0 \leq \rho'\bar{d}(u)|S+d(u)|S+w_0$$

u cannot switch OFF in $SD(S, \rho')$. Consider $u \in V \backslash S$ such that $n_u(S) > 0$.

$$-\rho\overline{d}(v)|S-d(v)|S-w_0 \stackrel{-n_v(S) \geq n_u(S) > 0}{\geq} \rho\bar{d}(u)|S+d(u)|S+w_0 \stackrel{\rho' \leq \rho < 0}{\Longrightarrow} -\rho'\overline{d}(v)|S-d(v)|S-w_0 \geq \rho'\bar{d}(u)|S+d(u)|S+w_0$$

u cannot switch ON in $SD(S, \rho')$ $\Box$.

Corollary 1 *If the first $k \geq 0$ switches of ρ-annealing are OFFS, then $SD(V, < -2N)$ switches OFF the same units in the same sequence of k switches.*

Corollary 1 suggests that ρ-annealing and $SD(V, < -2N)$ are very similar upto a point. Experiments (see Section 5) confirm this similarity; their closer analysis also reveals what characterizes the point at which they diverge. A simple example illustrates when Lemma 4 is inapplicable. Let the graph underlying HcN be a 4-cycle with the edges $(v_1, v_2), (v_2, v_3), (v_3, v_4)$, and (v_4, v_1). In the state $\{v_1, v_2, v_3\}$, with $\rho = -1$, v_4 will switch ON but with $\rho = -2$, v_1 or v_3 will switch OFF. A similar situation can occur during the operation of $SD(V)$ and ρ-annealing in a graph, causing the divergence.

Attractive features of ρ-annealing. Though $SD(V, < -2N)$ and ρ-annealing are very similar in behavior and in MAX-CLIQUE approximation performance (see Section 5), ρ-annealing holds some advantages over $SD(V, < -2N)$—the somewhat simpler dynamics. First, the similarity of their MAX-CLIQUE approximation performance notwithstanding, ρ-annealing almost always performs just slightly better (see Section 5). We have no explanation as yet for this curious phenomenon.

Second, on compressible graphs drawn from a distribution based on Kolmogorov complexity ideas too complicated to describe here, ρ-annealing occasionally outperforms $SD(V, < -2N)$ significantly [23]. On 6 out of 50 400-vertex compressible graphs to which both dynamics were applied, the clique sizes found by $SD(V, < -2N)$ and ρ-annealing were $(40, 58), (14, 20), (56, 42), (17, 29), (50, 70)$, and $(58, 76)$. $SD(V, < -2N)$ outperformed ρ-annealing once; ρ-annealing outperformed $SD(V, < -2N)$ 5 times.

Third, consider the optimization problem: given k and a graph G, find a sufficiently large k-degree-ratio set in G. This problem is a generalization of MAX-CLIQUE and may arise naturally in applications. It may also arise as a further relaxation of MAX-CLIQUE. The traditional relaxation of MAX-CLIQUE, motivated by its likely intractability, is to seek a sufficiently large, rather than largest, clique. A further relaxation of MAX-CLIQUE is to seek a "near" clique. Tractability is one motivation; perhaps a more important one is when the objective of obtaining a large vertex set outweighs the need for it to be a clique (near cliques are larger than their corresponding cliques). Section 5 (Table IV) gives precise empirical results on this statement. Our notion of k-degree-ratio sets is one characterization of a near-clique. ρ-annealing is attractive because the sequence of its stable states is characterizable as a sequence of $|\rho|$-degree-ratio sets, for any desired ρ annealing schedule. Even when $SD(V, -2N)$ makes identical switches to ρ-annealing, it is not clear which states in the sequence represent k-degree-ratio sets. Because of this ability to generate a sequence of $|\rho|$-degree-ratio sets for any sequence of decreasing $|\rho|$'s, ρ-annealing may have applications to hierarchical clustering in pattern recognition via graph-theoretic methods (see [8]).

Running Time. It is known that Hopfield networks with $|w_{ij}| = \{0, 1\}$ converge in $O(N^2)$ unit-switches (see [25], Chapter 2, Theorem 2.3). This result thus also applies to the 0/1 weights network equivalent to HcN [13]. This equivalence however breaks down for arbitrary $\rho < 0$. Therefore, here, we explicitly prove the same result for HcN, for arbitrary $\rho < 0$. Rewriting (3):

$$E(S) = -w_0|S| - \frac{1}{2}\sum_{v \in S}[\rho\overline{d}(v)|S + d(v)|S] \tag{7}$$

From (6,7),

$$\Delta E_v(S) \;=\; \begin{cases} |\rho|\overline{d}(v)|S - d(v)|S - w_0 & \text{if } v \in V \backslash S \text{ switches ON} \\ \rho\overline{d}(v)|S + d(v)|S + w_0 & \text{if } v \in S \text{ switches OFF} \end{cases} \tag{8}$$

Let $\rho = -\frac{x}{y}$ and for convenience choose $w_0 = \frac{1}{2y}$.

$$v \in V \backslash S \text{ switches ON} \Rightarrow |\rho|\overline{d}(v)|S - d(v)|S \leq 0 \Rightarrow \Delta E_v(S) \leq \tfrac{-1}{2y}$$

$$v \in S \text{ switches OFF} \Rightarrow \rho\overline{d}(v)|S + d(v)|S < 0 \Rightarrow \frac{yd(v)|S - x\overline{d}(v)|S}{y} < 0 \Rightarrow \Delta E_v(S) \leq \tfrac{-1}{2y}$$

The maximum possible energy E_{max} is $E(V)$ when G is the empty graph. That is, $E_{max} = \frac{x\binom{N}{2}}{y} - \frac{N}{2y}$. The minimum possible energy E_{min} is $E(V)$ when G is the complete graph K_N. That is, $E_{min} = -\binom{N}{2} - \frac{N}{2y}$. Since $E_{max} - E_{min}$ is $O(N^2)$ and the minimum energy reduction caused by a unit-switch is $|\Delta E_v(S)| \geq \frac{1}{2y}$, convergence is in $O(N^2)$ unit-switches.

From the above analysis, with geometric annealing, ρ-annealing converges within $O(N^2 \log N)$ unit-switches. This bound is not tight however. First, the $O(N^2)$ upper bound on running time of SD does not appear tight. Experimentally, SD has always converged in $O(N)$ unit-switches. Second, our empirical observations are that almost all ρ-annealing unit-switches involve switch OFFs. We therefore strongly conjecture that ρ-annealing converges in $O(N)$ unit-switches and list it currently as an open problem. These empirical observations suggest that ρ-annealing is quite efficient.

3.3 Stochastic Steep Descent

Stochastic Steep Descent (SSD) is a stochastic variant of SD. In SSD, the deterministic moves of SD are replaced by *energy descent* moves that are stochastic but favor the steepest direction. The motivation is, with initial state V, to improve upon the already good optimization performance of $SD(V)$ (see Section 5) without a significant risk of worsening it. The unit to be updated is selected via a probability distribution P that has zero probability of "up-hill" moves and favors large (steepest-descent like) decreases in energy. Specifically, let $C(t) \equiv \{i | \Delta E_i(t) < 0\}$.

$$P[S_i \text{ is switched at time } t] \quad := \quad \begin{cases} 0 & \text{if } \Delta E_i(t) \geq 0 \\ \delta[\Delta E_i(t)] & \text{otherwise} \end{cases} \tag{9}$$

$$\sum_{i \in C(t)} \delta[\Delta E_i(t)] \;=\; 1 \tag{10}$$

(10) ensures that exactly one unit is switched. Our choice of δ (hence P) that approximates SD is the linear distribution, represented by

$$\delta[\Delta E_i(t)] = \frac{\Delta E_i}{\sum_{j \in C(t)} \Delta E_j} \tag{11}$$

The probability of switching a unit is proportional to the amount of energy the switch decreases. SSD performs only gradient descent moves for two reasons: (1) to approximate SD and (2) because any serial-update gradient descent update scheme on HcN converges in $\leq 2N$ unit switches (Lemma 2). Non gradient-descent heuristics like simulated annealing are *much* slower. The idea behind SSD is that different runs on the same input will produce different solutions and because it approximates SD, it is expected to produce at least one solution that is better than SD in a small number of runs. Let $SSD_{max}(V_0, i)$ denote i runs of SSD on the same graph, with V_0 as input for each run. The best clique found is chosen. In Section 5, we will see that $SSD_{max}(V, N)$ performs significantly better than $SD(V, < -2N)$ and $SSD_{max}(V, 1)$, validating our ideas.

Analysis of SSD. Though one main feature of SSD is stochastic approximation of steepest descent which is best exhibited when the input is V (see Section 5 for good performance of $SSD_{max}(V, N)$), for simplicity, we restrict most of our analysis to the case when the input is $\emptyset$. $SSD(\emptyset)$ starts from the initial state $\emptyset$—all units OFF—and sequentially turns units ON that are adjacent to every unit in the previous state. At any given time, all candidate (i.e. energy-reducing) units are equiprobable to switch. Thus this dynamics is exactly the true (i.e fully local) asynchronous Hopfield dynamics, which is an implementation advantage in its favor over our other discrete dynamics (including $SSD(V)$).

The following lemma shows that if sufficiently-many runs are employed, stochastic steep descent will find the largest clique with probability 1. Let $\omega(G)$ denote the size of the largest clique in G.

Lemma 5 $Pr[\lim_{i \to \infty} SSD_{max}(V, i) = \omega(G)] \to 1$. $Pr[\lim_{i \to \infty} SSD_{max}(\emptyset, i) = \omega(G)] \to 1$.

Proof: Every $\omega(G)$-clique is reachable by some sequences of $\omega(G)$ switch ON's from the initial state ϕ and hence the probability of retrieval is $\geq$ a fixed $\epsilon > 0$. The result follows because the runs are independent trials. Similarly, every $\omega(G)$-clique is reachable by some sequences of switch OFFs and ONs from the initial state V and hence has a probability of retrieval $\geq \epsilon > 0$ $\Box$.

It is worthy of note that $\emptyset$ and V are the only two "universal" initial states from which all stable states are reachable by energy-reducing sequences of switches. Consider any other state S. Consider any graph with two connected components: S and $V \backslash S$. No stable state in $V \backslash S$ is reachable from the initial state S.

We now put better bounds on the expected number i of runs in which a clique of given size may be found by $SSD_{max}(\emptyset, i)$. Define a k-sequence as the sequence of the first k vertices output by one run of $SSD(\emptyset)$. A k-sequence is a k-clique, output in a particular order. Assume that G has a k-clique. For all i,

$$PR[i^{th} \text{ run of SSD}(\emptyset) \text{ involves} \geq k \text{ steps } \textit{and} \text{ outputs a given } k\text{-sequence}] \geq \frac{1}{n(n-1)(n-2)\ldots(n-k+1)} \tag{12}$$

Inequality (12) arises from the fact that the probability of obtaining vertex v_{i1} in the first step is $\frac{1}{n}$; the probability of obtaining vertex v_{i2} in the second step is $\geq \frac{1}{n-1}$ and so on. Because there are $k!$ k-sequences of a given k-clique,

$$PR[i^{th} \text{ run of SSD}(\emptyset) \text{ involves} \geq k \text{ steps } \textit{and} \text{ outputs a given } k\text{-clique in the first } k \text{ steps}] \geq \frac{k!}{n(n-1)(n-2)\ldots(n-k+1)} = \frac{1}{\binom{n}{k}} \tag{13}$$

$$PR[i^{th} \text{ run of SSD}(\emptyset) \text{ outputs a clique of size} \geq k] \geq \frac{\text{number of } k\text{-cliques in } G}{\binom{n}{k}} \tag{14}$$

Note that the R.H.S. of (14) is also the probability that k vertices selected uniformly at random form a clique. However, because SSD exploits the structure of the graph and the "randomly-generate and test" algorithm does not, this bound is weak for SSD. In practice, SSD is expected to perform much better. From (14), the expected number of runs i so that $SSD_{max}(\emptyset, i)$ finds a clique of size $\geq k$ is:

$$\leq \frac{\binom{n}{k}}{\text{number of } k\text{-cliques in } G} \tag{15}$$

We now refine the above SSD analysis for p-random graphs, exploiting some structure of the graphs to obtain a slightly better bound. A p-random n-vertex graph is one in which each of the $\binom{n}{2}$ vertex-pairs is connected by an edge, independent of other edges, with probability p. The expected degree of every vertex is $p(n-1)$ and the expected number of k-cliques is $\binom{n}{k}p^{\binom{k}{2}}$ [6, 33]. To simplify analysis, for fixed p and sufficiently large n we assume that the degree of each vertex is $\leq n\frac{1+p}{2}$ and the number of k-cliques is exactly $\binom{n}{k}p^{\binom{k}{2}}$. By analysis similar to (10-13),

$$PR[i^{th} \text{ run of SSD}(\emptyset) \text{ involves} \geq k \text{ steps } \textit{and} \text{ outputs a given } k\text{-clique in the first } k \text{ steps}] \geq \frac{k!}{n(n\frac{1+p}{2}-1)(n\frac{1+p}{2}-2)\ldots(n\frac{1+p}{2}-k+1)} \geq \frac{1}{(\frac{1+p}{2})^{k-1}\binom{n}{k}} \tag{16}$$

By substituting for number of k-cliques, we obtain:

$$PR[i^{th} \text{ run of SSD}(\emptyset) \text{ outputs a clique of size} \geq k] \geq \frac{\binom{n}{k}p^{\binom{k}{2}}}{(\frac{1+p}{2})^{k-1}\binom{n}{k}} = \frac{p^{\binom{k}{2}}}{(\frac{1+p}{2})^{k-1}} \tag{17}$$

The expected number of runs i so that $SSD_{max}(\emptyset, i)$ finds a clique of size $\geq k$ is thus:

$$\leq \frac{(\frac{1+p}{2})^{k-1}}{p^{\binom{k}{2}}} \tag{18}$$

The improvement in (18) over substituting for the number of k-cliques in (15) is a factor of $(\frac{1+p}{2})^{k-1}$. It is easy to check, however, that when $p = 0.5$ (the uniform distribution on graphs) and $k \approx 2\log_2 n$ (the expected maximum clique size in 0.5-random graphs; see Section 5), the bound of (18) grows faster than any polynomial. First, it is clear that the bound is not tight. The advantage is simplicity of analysis: the graph is *fixed* before the first run and the only random variables are SSD's stochastic decisions. A tighter analysis would need to consider the edges of the graphs as random variables that get individually fixed during SSD's evolution; this complicates matters. Second, it is even unclear that a sufficiently good bound is possible. There is an interesting conjecture by Jerrum [24] which states that, for any $\epsilon > 0$, there is no polynomial time algorithm that with probability greater than half can find a clique of size $(1+\epsilon)\log_2 n$ in a $\frac{1}{2}$-random graph. Thus, though the simplest heuristic can find a clique whose size is half of optimal in a random graph, the existence of a polynomial time algorithm that can find a significantly better one is doubted.

A brief note about SSD vs backtracking search: the latter requires memory to remember the visited stable states and control to coordinate their visitation; the former requires memory only to remember the largest stable state so far. Consequently, SSD is much better suited for parallel distributed neural implementation.

4 Continuous Descent Dynamics

Continuous Hopfield Dynamics. The first continuous dynamics on HcN that we will use to approximate MAX-CLIQUE is the continuous energy-descent dynamics (CHD) of Hopfield [16] (see also [14]). It is described, in one form, by the system of N coupled nonlinear differential equations:

$$\frac{dS_i}{dt} = -S_i + g_\lambda(w_0 + \Sigma_j w_{ij} S_j) \tag{19}$$

The fixed points satisfy $S_i = g_\lambda(w_0 + \Sigma_j w_{ij} S_j)$. CHD minimizes the energy function of (4) and so on HcN—for sufficiently large λ—the fixed points approximate characteristic vectors of maximal cliques of the underlying G_N. The choice of g as a sigmoid is also convenient for analog circuit implementations. The sigmoid is also related to the Boltzmann distribution; the significance of which is briefly reviewed in the next paragraph. CHD is a special case of Mean Field Annealing (see below). CHD is realizable in simple analog circuit implementations [16] (also see [14]). It can also be solved numerically in fully parallel form, as illustrated below.

$$\bar{S}_{n+1} := \bar{S}_n + \gamma(-\bar{S}_n + \bar{g}_\lambda(W\bar{S}_n + (w_0))) \tag{20}$$

where $\bar{g}(\bar{x})$ is notational shorthand for $(g(x_i))$. γ is the step size. One iteration of (20) can be done in N g-evaluation, $\Theta(N^2)$ multiplication, and $\Theta(N)$ addition steps with 1 processor; 1 g-evaluation, $\Theta(N)$ multiplication, and $\Theta(NLogN)$ addition steps with N processors; and 1 g-evaluation, $\Theta(1)$ multiplication, and $\Theta(LogN)$ addition steps with N^2 processors.

Mean Field Annealing. The fourth dynamics that we will use to approximate MAX-CLIQUE is Mean Field Annealing (MFA) [5, 34], an efficient approximation to Simulated Annealing (SA) [28]. When applied to a Hopfield Network, MFA can be viewed as a generalization of CHD in which the parameter $T = \frac{1}{\lambda}$ is annealed (varied).

We review a Stochastic Hopfield Network (SHN); then its Mean Field approximation; then MFA. SHN (see [14]) is a stochastic version of DHN in which the units are discrete, but the update rule is stochastic:

$$Prob[S_i = 1] = g_{\frac{1}{T}}(w_0 + \Sigma_j w_{ij} S_j) \tag{21}$$

where the choice of update function as $g_{\frac{1}{T}}$ makes the energy Boltzmann-distributed. Units are updated until thermal equilibrium is reached. That is, the averages $< S_i >$ stop changing. SA is a generalization of SHN in which T is annealed.

The Mean Field approximation to (refeq:MFA1) equates the average $< S_i >$ in terms of the averages $< S_j >$:

$$< S_i >= g_{\frac{1}{T}}(\Sigma_j w_{ij} < S_j >) \tag{22}$$

(22) simplifies analysis of SHN and SA. As important, with continuous units representing $< S_i >$, CHD ((19)) evolves to a fixed point satisfying (22). Thus CHD, an efficient energy-descent-only dynamics, is a Mean Field approximation to the relatively inefficient SHN dynamics. MFA is a generalization of CHD in which $T = \frac{1}{\lambda}$ is annealed. Thus, MFA is an efficient Mean Field approximation to SA.

In MFA we (1) start at a high temperature T (small λ), (2) reach a fixed point by CHD, and (3) decrease T. We repeat Steps 2,3—using the fixed point of the previous Step 2 as the initial state of the next Step 2—until T is sufficiently small. The prescription for decreasing T is called the annealing schedule. A parallel numerical implementation of MFA is:

$T \leftarrow T_0$; Initialize $\bar{S}_0$
repeat
  repeat
    $\bar{S}_{n+1} := \bar{S}_n + \gamma(-\bar{S}_n + \bar{g}_\lambda(W\bar{S}_n + (w_0)))$
  until $\bar{S}_{n+1}$ is a fixed point;
  Decrease T; $\bar{S}_0 \leftarrow \bar{S}_{n+1}$
until T is sufficiently small

MFA

In practice, the inner repeat loop may be iterated for a fixed number of iterations (as we have done; see Section 5) or stopped "close" to a fixed point, namely, when $|\bar{S}_{n+1} - \bar{S}_n| \leq \delta$, where δ is a pre-specified threshold.

5 Experiments and Results

In this section, we apply the various HcN dynamics to MAX-CLIQUE and empirically evaluate their approximation performance.

5.1 Test Graphs

We test performance on two types of graphs: p-random graphs and k random cliques graphs.

p-random graphs. p-random graphs, defined in Section 3.3, represent a large uniform class of easily replicable (in statistical sense) graphs. Experimental results on p-random graphs may be interpreted as empirical estimates of average-case performance of our algorithms, making average-case

comparisons with other algorithms possible. Results on 0.5-random graphs, the uniform distribution on graphs, apply to "almost all" graphs. p-random graphs are also useful because several surprising theoretical results hold for them. One such result, discovered and proved by Matula (see [33], pg 76), very accurately pinpoints the size of the maximum clique in a p-random graph. Let

$$d(N) \equiv 2\log_{\frac{1}{p}} N \; - \; 2\log_{\frac{1}{p}} \log_{\frac{1}{p}} N + 1 + 2\log_{\frac{1}{p}} \frac{e}{2}$$

Matula's theorem says that, for large N, the maximum clique size in almost all p-random N-vertex graphs is either $\lfloor d(N) \rfloor$ or $\lceil d(N) \rceil$. An equally important related result [6] is that the size of the smallest maximal clique in a p-random graph is almost surely $\frac{d(N)}{2}$, that is, half that of the maximum clique. The computational significance of this result is that *any* algorithm that returns only maximal cliques (e.g. all algorithms in this paper) will almost surely find a clique within factor two of optimal in a p-random graph. On the flip side, however, there is the conjecture of Jerrum refered to in Section 3.3 which states that though factor two maximum clique approximation on p-random graphs is easy, finding even a slightly larger clique may be hard.

Both these results and Jerrum's conjecture are asymptotic (i.e. for sufficiently large N). We have experimentally determined (see Table I) that on small graphs (N between 100–400), $d(N)$ is a poor estimate of maximum clique size (and $\frac{d(N)}{2}$ a poor estimate of smallest maximal clique size) when p is large (e.g. $p = 0.9$). We choose to estimate the maximum clique size and smallest maximal clique size directly from the expected number of maximal cliques in p-random N-vertex graphs [6]:

$$E_{N,p}[k] = \binom{N}{k} p^{\binom{k}{2}} (1 - p^k)^{n-k} \tag{23}$$

First, the expected sizes k_1 and k_0 of maximum clique and smallest maximal clique respectively are obtained by tabulating the distribution $E_{N,p}[k]$ (which is binomial-like) and noting the k for which the values transition from ≈ 1 to $<< 1$ and from $<< 1$ to ≈ 1 respectively. These numerical estimates are much more accurate than $d(N)$ on the graphs in our experiments. Our experimental results are consistent with them. Table I also shows that, notwithstanding Jerrum's conjecture, several dynamics perform significantly better than factor two of optimal on 100- to 400-vertex p-random graphs.

k random cliques graphs. These graphs are generated by generating k cliques of varying size at random and taking their union. Such graphs have a wide range of clique sizes—much wider than in p-random graphs. This suggests that such graphs are "hard" for approximating MAX-CLIQUE and may separate the poor algorithms from the good ones. Our experimental results support this suggestion (see Tables I and II; especially SD($\emptyset$) and SSD($\emptyset, 1$) vs SD(V)).

For ease-of-programming, we used the following generation algorithm:

1. $G \leftarrow \emptyset$ (Start with empty graph)
2. for $i := 1$ to k do
   (a) Pick a random number p between 0 and $\frac{1}{2}$
   (b) Generate an N-bit vector as a sequence of N Bernoulli trials(probability of each bit being 1 independently equals p).
   (c) Interpret this vector as a subset S of $\{1, 2, \ldots, N\}$ of expected size pN
   (d) Interpret S as a clique
   (e) $G \leftarrow G \cup S$ (Add clique S to graph G)

5.2 Results

Table I summarizes the experimental approximation performance of the various HcN dynamics on MAX-CLIQUE. The numbers are the clique sizes returned by the various algorithms, averaged over the number of graphs indicated in the header rows. Column 2 ($|V|$) indicates the graph order (number of vertices). The last two columns (labeled EMi; EMa) are the expected sizes of the smallest maximal clique and maximum clique respectively and are estimated by numerically tabulating the distribution of (23), as described earlier. The third last column is $d(N)$—Matula's estimate. These three columns are meaningful only for p-random graphs.

Place Table I here

On p-random graphs, the performance of all algorithms vary little from their means. Table III reports these variations in detail. Therefore the averages reported in Table I are accurate estimates of performance. On k random cliques graphs, the clique sizes returned by the algorithms are significantly dependent on the graph instance. Therefore, the Table I averages are not good estimates of absolute performance. Table II reports the performance on individual 400-vertex graphs. The first column is the lower bound on the maximum clique size obtained by recording the largest set used while generating each k random cliques graph.

Place Tables II and III here

Parameter Settings. The discrete dynamics SD($\emptyset$), SD(V), SSD($\emptyset, 1$),SSD($V, 1$),SSD($\emptyset, N$), and SSD(V, N) require only that $\rho < -N$. The parameter settings for the remaining dynamics ρ-annealing, CHD, and MFA are important. Their experimental settings were as follows.

The ρ-annealing schedule was geometric: $\boxed{\rho_n = 2\rho_{n-1};\ \ \rho_0 = -\frac{1}{8}}$ and operated until $\rho_n < -N$ inclusive. This schedule's main motivation is speed: slowing the schedule is unlikely to affect the performance significantly given the connection between ρ-annealing and $SD(V)$. The bias w_0 was chosen equal to 0.1.

CHD was operated at $\rho = -4N$, $w_0 = \frac{|\rho|}{4}$, $T = \frac{1}{\lambda} = 1, \gamma = 0.1$, and # iterations = N. The initial state was $S(0) = (0.5 + \delta)^N$ where δ is uniform random noise in $[-0.05, 0.05]$. We briefly explain the choice of w_0. First, w_0 must be greater than 0 because $w_0 = 0$ admits the empty set as a stable state which in the past CHD has frequently retrieved [20]. Furthermore, for efficiency reasons, it is preferable to choose w_0 reasonably large under the constraint that stable states remain unchanged (i.e. exactly the set of maximal cliques). We first note that for *any* positive w_0, a non-maximal clique cannot be a stable state. The only possibility then is that overly large w_0 may admit non cliques as stable states. It is easy to check that choosing $w_0 = \frac{|\rho|}{4}$ is sufficient to avoid this possibility. Consider any pair v_1, v_2 of vertices in a stable state. Since $\rho = -4N$, even if v_1 and v_2 are each adjacent to all the remaining $N - 2$ vertices and all vertices are ON, the input to v_1 and v_2 is nonnegative if and only if v_1 and v_2 are adjacent.

The initial state to MFA was the same as to CHD. ρ and w_0 were also the same. For every fixed T, MFA was operated with $\gamma = 0.1$ and # iterations-per-T $= N$. The MFA annealing schedule was geometric: $\boxed{T_n = a_{n-1}T_{n-1};\ \ T_0 = \frac{2}{3}(1-p)N|\rho|}$ where $a_n = 0.9; n \leq 3$, $a_n = 0.5; n > 3$. The schedule was operated until $T_n = 1$ inclusive. The initial temperature T_0 was arrived at by the following approximate discrete analysis (in crude analogy with instability analysis in dynamical systems). Assume that the initial network state is $S(0) = 0.5^N$. Our goal is to find the maximum temperature T to achieve a pre-specified desired change in every unit's state after one iteration. In our case, our pre-specified desired change was $\Delta S_i = 0.01$. Consider a p-random N-vertex graph. From (20), noting that the degree of every vertex is approximately pN, our parameter settings,

and the initial state $S(0)$, for every unit i:

$$\Delta S_i(a) \approx 0.1(-0.5 + \frac{1}{1+e^{\frac{a}{2}}}) \tag{24}$$

where $a = \frac{(1-p)|\rho|N - w_0}{T}$. Numerically, we have noted that for $a \in (1,2)$, $\Delta S_i(a) \in (0.01, 0.02)$. Therefore any $a \in (1,2)$ would satisfice. Choosing $a = \frac{3}{2}$, a somewhat arbitrary choice, and dropping w_0 as insignificant, gives $T = \frac{2}{3}(1-p)|\rho|N$. For k random cliques graphs, we have used the same setting but with $p = \frac{1}{2}$. The MFA settings are robust to graph size scaling (see Table I).

Analysis. All dynamics exhibit trade-offs between implementation cost, running time, and approximation performance. CHD and MFA are implementable in analog circuits. By our choice of parameters, they require $\Theta(N)$ and $\Theta(N \log N)$ iterations respectively. Though the SD and SSD dynamics also involve $\Theta(N)$ unit-switches, on a sequential machine, one unit-switch can be implemented significantly faster than one iteration of CHD or MFA. Consequently SD and SSD are significantly faster than CHD and MFA on sequential machines. ρ-annealing holds the same advantage to a slightly lesser extent. On a parallel distributed machine, one unit-switch of SD($\emptyset$) and SSD($\emptyset, 1$) is faster than one unit-switch of SD(V) because the latter requires one global computation and the former does not. SSD($\emptyset, 1$) may in fact be implemented as the true discrete asynchronous Hopfield dynamics and SSD($\emptyset, N$) also shares this advantage to a slightly lesser extent but significantly improves on the approximation performance. SD(V) however exhibits significantly better approximation performance than either SD($\emptyset$) or SSD($\emptyset, 1$). CHD is both slower and exhibits slightly poorer performance than SD(V). On a semi-loaded SUN Sparc-station, the time to find a clique is roughly, depending on N (100 to 400), 30-90 seconds for one run of the discrete dynamics, 5-30 minutes for CHD, and an hour or so for MFA. Though all dynamics are inherently parallel, parallel implementation of MFA should provide the greatest speed-up.

We now focus on the approximation performance of these dynamics. From Table I, MFA and SSD(V, N) are the narrow but clear winners in most tests. On 0.9-random graphs, MFA is the clear sole winner by a significant margin. A slower annealing schedule should further improve MFA's performance (see paragraph: **Parameter Analysis** below). Somewhat surprisingly, SD(V))—a truly simple heuristic—performs very well, better than CHD, and almost as well as the winners. Though the performance of SD(V) and ρ-annealing is expected to be almost the same (Lemma 1), somewhat curiously the latter always performs just slightly better than the former. SD(Φ) performs the worst.

We now compare the dynamics pair-wise to note the effects of certain features.

- Comparison of $SD(\emptyset)$ and $SD(V)$ columns shows the effect of the initial state: $SD(V)$ is significantly better.
- Comparison of CHD and MFA columns shows the improvement due to annealing: MFA is significantly better.
- Comparison of $SSD(V, 1)$ and $SSD(V, N)$ columns shows the improvement due to N runs: $SSD(V, N)$ is significantly better.
- Comparison of $SD(\emptyset)$ and $SSD(\emptyset, 1)$ columns shows the improvement due to randomization: $SSD(\emptyset, 1)$ is always moderately better.

From Table III, we may note the following features: (a) the variances in the performance of the algorithms are small and the distributions are binomial-like, (b) the variances increase markedly

as p is increased, (c) the variances of SSD$(\emptyset, N)$ and SSD(V, N) are significantly lower than the rest (because multiple runs reduce the variance), (d) more detailed evidence that ρ-annealing and $SD(V)$ perform similarly.

Parameter Analysis. We single out two parameters, based on their perceived importance, to study the effects of their settings on solution quality: (a) the number of runs in $SSD(V, N)$ and (b) the annealing schedule in MFA. The experiments are rudimentary but give clear indications. On a fresh set of 15 0.5-random graphs, $SSD(V, 20)$ retrieved an average clique size of 7.87; $SSD(V, 100)$ an average clique size of 8.67. On the 50 0.5-random graphs of Table I, $SSD(V, 20)$ retrieved an average clique size of 8.12; $SSD(V, 100)$, as noted in Table I, retrieves an average clique size of 8.60. On the 50 0.9-random graphs of Table I, $SSD(V, 20)$ retrieved an average clique size of 27.64; $SSD(V, 100)$, as noted in Table I, retrieves an average clique size of 28.76. The MFA annealing schedule was geometric: $T_n = a_{n-1}T_{n-1}; T_0$ as before. On a fresh set of 15 0.909-random graphs, MFA with $a_i = 0.5$ retrieved an average clique size of 31.4; MFA with $a_i = 0.8$ retrieved an average clique size of 31.87.

The following MFA parameter settings deserve further study: initial temperature $T(0)$, annealing schedule, number of iterations per T. The main objective of this paper with regards to MFA has however been accomplished: it is shown, with a reasonable choice of parameters, that MFA is tied for the best performance. Also, on 0.9-random graphs, the MFA performance is already near-optimal and the sole winner.

k-degree-ratio sets. The maximum clique size in p-random graphs is small. In certain applications, it may be desirable to retrieve larger sets at the expense of relaxing the clique condition to a "near" clique (in particular to a k-degree-ratio set). One attractive feature of ρ-annealing, mentioned earlier, is its ability to retrieve a sequence of k-degree-ratio sets for any desired sequence of increasing k's. This processing is not possible with other dynamics. Table IV reports on the sizes of the k-degree-ratio sets returned by ρ-annealing as intermediate stable states on the p-random graphs of Table I. We get a good appreciation of the trade-off (size gain vs degree of relaxation of clique property). Of special interest is the existence of an interesting threshold in the trade-off. Starting from right, each preceeding (i.e. next left) column represents a factor of two relaxation (i.e. allowable $\frac{d(v)|S}{\overline{d}(v)|S}$ is multiplied by two) and the improvement in size is noted. The threshold occurs (especially $N = 400$) in going from the column $\frac{d(v)|S}{\overline{d}(v)|S} = 2$ to $\frac{d(v)|S}{\overline{d}(v)|S} = 1$: the size (for $N = 400$) improves from 42 to 262. For $p = 0.9$ graphs, the threshold has shifted to the right (i.e. when the constraints are tighter).

< Place Table IV here >

6 Conclusion

MAX-CLIQUE is a classic graph optimization problem that is NP-hard even to approximate well. For this and related reasons, it is a problem of considerable interest in theoretical computer science. MAX-CLIQUE also has several real-world applications.

In this paper, we have encoded MAX-CLIQUE in a special case of the Hopfield Network and approximately solved it via several discrete (deterministic and stochastic) and continuous energy-descent dynamics. We have noted that some of these dynamics emulate well-known MAX-CLIQUE heuristics, whereas others (e.g. MFA and ρ-annealing) arise from the neural network connection. We have theoretically characterized the properties of several dynamics in the context of their application to MAX-CLIQUE.

Our detailed experiments on p-random and k random cliques graphs have shown that mean field annealing and stochastic steep descent perform the best and $SD(\emptyset)$, the "naive" greedy dynamics, performs the worst. The other dynamics have their advantages.

7 Acknowledgements

The author thanks Dr. Kenneth Regan for discussions leading to the suggestion of graphs containing large cliques (which led to k random cliques graphs).

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *The Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science*, page to appear, 1992.

[2] G. Augustson and J. Minker. An analysis of some graph-theoretical clustering techniques. *Journal of the Association of Computing Machinery*, 17(4):571–588, October 1970.

[3] E. Balas and C.S. Yu. Finding a maximum clique in an arbitrary graph. *SIAM Journal on Computing*, 15(4), November 1986.

[4] D.H. Ballard, P.C. Gardner, and M.A. Srinivas. Graph problems and connectionist architectures. Technical report, Department of Computer Science, University of Rochester, Rochester, NY, 1987.

[5] G. Bilbro, R. Mann, T.K. Miller, W.E. Snyder, D.E. Van den Bout, and M. White. Optimization by mean field annealing. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1, pages 91–98, San Mateo, 1989. (Denver 1988), Morgan Kaufmann.

[6] B. Bollobás and P. Erdös. Cliques in random graphs. *Proc. Camb. Phil. Soc.*, 80:419–427, 1976.

[7] P. Crescenzi, C. Fiorini, and R. Silvestri. A note on the approximation of the maxclique problem. *Information Processing Letters*, 40(1):281–298, October 1991.

[8] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, New York, 1973. Chapter 6: Unsupervised Learning and Clustering (Section 6.11).

[9] U. Freige, S. Goldwasser, L. Lovasz, S. Safra, and M. Szegedy. Approximating clique is almost NP-complete. In *Symposium on Foundations of Computer Science*, pages 2–12, 1991.

[10] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[11] G.H. Godbeer, J. Lipscomb, and M. Luby. On the computational complexity of finding stable state vectors in connectionist models (Hopfield nets). Technical report, Department of Computer Science, University of Toronto, Toronto, Ontario, 1988.

[12] J.R. Griggs. Lower bounds on the independence number in terms of the degrees. *Journal of Combinatorial Theory, Series B*, 34:22–39, 1983.

[13] T. Grossman and A. Jagota. On the equivalence of two Hopfield-type networks. In *IEEE International Conference on Neural Networks*, 1993. Submitted to.

[14] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[15] J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences, USA*, 79, 1982.

[16] J.J. Hopfield. Neurons with graded responses have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences, USA*, 81, 1984.

[17] R. Horaud and Skordas T. Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1168–1180, 1989.

[18] A. Jagota. Applying a Hopfield-style network to degraded text recognition. In *International Joint Conference on Neural Networks*, volume 1, New York, 1990. San Diego, July, IEEE.

[19] A. Jagota. A new Hopfield-style network for content-addressable memories. Technical Report 90-02, Department of Computer Science, SUNY at Buffalo, Buffalo, NY, 1990.

[20] A. Jagota. Efficiently approximating max-clique in a Hopfield-style network. In *International Joint Conference on Neural Networks*, volume 2, pages 248–253, New York, June 1992. Baltimore, June, IEEE.

[21] A. Jagota. A Hopfield-style network with a graph-theoretic characterization. *Journal of Artificial Neural Networks*, 1992. To Appear.

[22] A. Jagota. Optimization by reduction to maximum clique. In *IEEE International Conference on Neural Networks*, New York, March 1993. San Francisco, IEEE. Submitted To.

[23] A. Jagota and K.W. Regan. Performance of max-clique heuristics under description-length weighted distributions. Technical Report 92-24, Department of Computer Science, SUNY at Buffalo, Buffalo, NY, September 1992. Available from author.

[24] M. Jerrum. The elusiveness of large cliques in a random graph. Technical Report CSR-9-90, Department of Computer Science, University of Edinburgh, September 1990.

[25] Y. Kamp and M. Hasler. *Recursive Neural Networks for Associative Memory.* Wiley, New York, 1990.

[26] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[27] R.M. Karp. The probabilistic analysis of some combinatorial search algorithms. In J.F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results*, pages 1–19. Academic Press, New York, 1976.

[28] S. Kirkpatrick, C.D. Gelatt Jr., , and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220, 1983.

[29] J.E. Lecky, O.J. Murphy, and R.G. Absher. Graph theoretic algorithms for the pla folding problem. *IEEE Transactions on Computer Aided Design*, 8(9):1014–1021, 1989.

[30] F. Lin and K. Lee. A parallel computation network for the maximum clique problem. In *First International Conference on Fuzzy Theory and Technology*, October 1992.

[31] A.K. Mackworth. Constraint satisfaction. In S.C. Shapiro, editor, *Encyclopedia of Artificial Intelligence, Second Edition*, pages 285–290, New York, 1992. John Wiley and Sons.

[32] H. Ogawa. Labeled point pattern matching by delaunay triangulation and maximal cliques. *Pattern Recognition*, 19(1):35–40, 1986.

[33] E.M. Palmer. *Graphical evolution.* Wiley, New York, 1985. Matula's theorem on page 76.

[34] C. Peterson and B. Söderberg. A new method for mapping optimization problems onto neural networks. *International Journal of Neural Systems*, 1:3–22, 1989.

[35] J. Ramanujam and P. Sadayappan. Optimization by neural networks. In *IEEE International Conference on Neural Networks*, volume 2, pages 325–332, New York, 1988. (San Diego 1988), IEEE.

[36] I. Shariv, T. Grossman, E. Domany, and A.A. Friesem. All-optical implementation of the inverted neural network model. In *Optics in Complex Systems*, volume 1319. SPIE, 1990.

[37] Y. Shrivastava, S. Dasgupta, and S.M. Reddy. Neural network solutions to a graph theoretic problem. In *Proceedings of IEEE International Symposium on Circuits and Systems*, pages 2528–2531, New York, 1990. IEEE.

Table I. Average size of retrieved clique by various algorithms,
on p-random graphs and k random cliques graphs

	Performance of Algorithms									Exp		
$\|V\|$	SD($\emptyset$)	SD(V)	ρ-A	SSD($\emptyset, 1$)	SSD($V, 1$)	SSD($\emptyset, N$)	SSD(V, N)	CHD	MFA	$d(N)$	EMi	EM
50 0.5-random graphs in each row												
100	6.34	7.98	8.06	6.48	6.42	8.36	8.6	7.44	8.5	9.7	4	10
400	8.30	9.88	10.34	8.44	8.24	10.8	11.04	9.16	10.36	12.94	5	13
50 0.9-random graphs in each row												
100	23.86	28.16	28.34	23.40	24.82	27.60	28.76	27.92	30.02	22.53	13	35
400	36.12	43.80	44.58	35.84	36.82	41.86	43.20	43.24	49.94	43.855	19	55
20 20-random-cliques graphs in each row												
100	35.45	48.45	48.8	42.85	44.45	50.35	50.9	45.9	49.7			
400	110.75	179.15	183.35	133.2	150.95	188.5	183.9	152.95	190.1			

Table II. Size of retrieved clique by various algorithms on
on the 20 '20 random cliques' 400-vertex graphs of Table I.

$\omega(G)$ L.B.	SD($\emptyset$)	SD(V)	ρ-A	SSD($\emptyset, 1$)	SSD($V, 1$)	SSD($\emptyset, N$)	SSD(V, N)	CHD	MFA
193	179	176	193	139	150	189	179	160	193
187	34	162	167	104	131	187	177	140	172
200	86	182	182	110	159	200	186	164	200
202	144	186	189	172	166	186	192	163	201
188	188	188	188	105	137	188	188	168	188
207	69	203	203	163	184	199	204	190	211
160	96	141	160	112	96	160	160	101	160
210	173	203	213	139	192	210	207	189	215
180	70	176	176	136	128	180	167	128	180
194	75	185	185	141	141	194	184	156	185
184	127	159	164	116	122	184	179	142	184
212	184	204	212	141	183	212	204	177	212
189	43	174	174	149	162	189	183	161	191
200	91	194	200	148	182	200	200	182	200
201	149	195	196	156	175	201	199	169	206
160	103	160	160	100	110	160	160	103	160
197	56	185	195	151	157	192	191	156	195
160	124	141	141	102	118	160	146	94	158
204	194	209	209	162	193	204	209	186	209
182	30	160	160	118	133	175	163	130	182

Table III a. Distributions of clique sizes found for 400-vertex 0.5-random graphs

Dynamics	6	7	8	9	10	11	12
$SD(\emptyset)$		10	22	11	7		
$SD(V)$			2	18	16	12	2
ρ-A			1	5	24	16	4
$SSD(\emptyset, 1)$	1	2	25	19	2	1	
$SSD(V, 1)$		10	21	16	3		
$SSD(\emptyset, N)$					10	40	
$SSD(V, N)$					1	46	3
CHD			9	27	11	3	
MFA				4	27	16	3

Table III b. Distributions of clique sizes found for 400-vertex 0.9-random graphs

Dynamics	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53
$SD(\emptyset)$			1	4	7	6	9	12	6	3	1	1												
$SD(V)$											1	3	7	9	16	7	3	3	1					
ρ-A											1	6	4	13	10	11	4	1						
$SSD(\emptyset, 1)$			1	3	9	10	10	8	4	3	2													
$SSD(V, 1)$	1		1	2	3	5	11	10	5	6	2	2	2											
$SSD(\emptyset, N)$											3	14	20	13										
$SSD(V, N)$													9	27	10	3	1							
CHD										2	1	7	7	9	9	10	5							
MFA																	1	1	7	10	14	8	7	2

Table IV. Stable state sizes obtained by ρ-annealing for different ρ and satisfying $\frac{d(v)|S}{\bar{d}(v)|S} \geq |\rho|$ on p-random graphs. The sizes are averaged over the 50 graphs of each row of Table I for the p-random graphs.

Test Graph Params		$\lvert\rho\rvert$, i.e. L.B. on $\frac{d(v)\lvert S}{\bar{d}(v)\lvert S}$								
N	p	$\frac{1}{4}$	$\frac{1}{2}$	1	2	4	8	16	32	64
100	0.5	100	99	66	23	12	9	8	8	8
400	0.5	400	400	262	42	18	12	10	10	10
100	0.9	100	100	100	99	77	43	34	28	28
400	0.9	400	400	400	400	400	330	105	59	44