ENS Paris

Laboratoire de Physique Statistique

Internship report

# Numerical Experiments on the Unit Distance Problem

*Author:*
Liang Qin
liang.qin@ens.fr

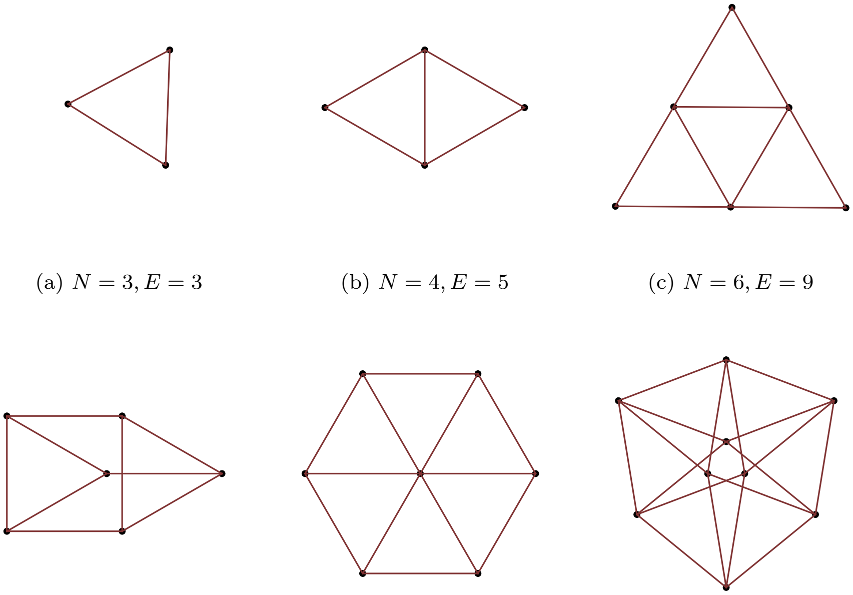
*Supervisor:*
Dr. Werner Krauth
werner.krauth@ens.fr

May 26, 2016

## Abstract

In this internship report I focus on a famous mathematical problem—the unit distance problem in two-dimensional euclidean space. Here the method of simulated annealing, several greedy algorithms, as well as the densest subgraph technique, constitute our main approaches to the graphs of ten to a thousand points. Though my results are no much better compared with constructions of lattices, the surprising beauty and regularity of outcomes of Monte Carlo simulated annealing(MCSA) lead us to seek for a universal way to obtain many-point solutions And at the same time, several facts imply the significance of the lattice—regarding symmetries and through lattice embedding test.

# Contents



(a) $N = 3, E = 3$

(b) $N = 4, E = 5$

(c) $N = 6, E = 9$

(d) $N = 6, E = 9$

(e) $N = 7, E = 12$

(f) $N = 9, E = 18$

Figure 1: The optimal configurations for the unit distance problem that can be drawn by hand.

# 1 Introduction

The mathematical definition of two-dimensional unit distance problem is

**Definition.** *Let $\boldsymbol{p}_i$, $i = 1, 2, \ldots, N$ be $N$ distinct points that can move freely on the plane without coincidence, $\boldsymbol{p}_i = (p_{i,x}, p_{i,y})$. The distance of two points is $dist(\boldsymbol{p}_i, \boldsymbol{p}_j) = \sqrt{(p_{i,x} - p_{j,x})^2 + (p_{i,y} - p_{j,y})^2}$. The function $U(N)$ is the maximum number of the pairs with distance exactly equal to one for $N$ points.*

We are not only interested in the value of $U(N)$ but also in the very placement of those $N$ points that achieves it. Several optimal graphs of small $N$ can be drawn by hand, as shown in Fig [1]. The study of this problem can revolve around: 1. specific values of $U(N)$ and the corresponding graphs; 2. the asymptotic behavior of $U(N)$ for $N \to \infty$.

A periodical version of unit distance problem is

**Definition.** *In a two-dimensional grid, all points can be represented by two non-parallel vectors $\boldsymbol{a}$*

*and* $\boldsymbol{b}$*, and a finite set of distinct points in its primitive cell* $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_n$*. Then any points can be written as* $\boldsymbol{p}_t + x\boldsymbol{a} + y\boldsymbol{b}$*,* $t \in \{1..n\}$*,* $x, y \in \boldsymbol{Z}$*. The connectivity of point* $u(\boldsymbol{p})$ *is the number of points with distance 1 to itself, as well as its average over the grid* $u = \frac{1}{2}\sum_{k=1}^{n} u(\boldsymbol{p})$*, and the density* $\rho = \frac{n}{|\boldsymbol{a}\times\boldsymbol{b}|}$*. Our goal is to maximize the connectivity as the density does not exceed* $\rho$*, denoted as* $u(\rho)$*.*

The equivalence of these two definitions is obvious, if we are only concerned about the asymptotic properties of $U(N)$ and $u(\rho)$, and in the condition that $\mathbf{a}\times\mathbf{b}$ keeps within $O(1)$: once $u(\rho)$ is attained, we cut a patch of area $S$ to obtain $N \sim \rho S$, $U \sim u(\rho)N = u(\frac{N}{S})N \sim u(N)N$, with a negligible loss at the margin; conversely, from $U(N)$, we have $u = \frac{U(N)}{N} = \frac{U(\rho S)}{\rho S} \sim \frac{u(\rho)}{\rho}$. Thus $U(x) \sim u(x)x$. The application of the lattice will be clear in section[2].

Two noticeable aspects of the problem are:

1. Any two points are forbidden to be identical to prevent the case that $\frac{N}{2}$ points lie at (0, 0) and the other half at (0, 1), which trivially renders $U = \frac{N^2}{4}$.

2. As for the variant of unit distance problem, when the dimension is equal or greater than 4, it also becomes trivial, placing $\frac{N}{2}$ points at $(\sin\theta_i, \cos\theta_i, 0, 0)$, the other half at $(0, 0, \sin\theta_i, \cos\theta_i)$, with $U = \frac{N^2}{4}$. Yet, the three-dimensional problem is still of interest.

This problem was first formulated by Paul Erdös in 1946. His initial upper bound, $U(N) \leqslant O(N^{\frac{3}{2}})$, is just a consequence of the fact that no two circles can intersect more than twice. Thus any unit distance graph contains no subgraph $K_{2,3}$(the graph with 5 vertices and edges 1-3, 1-4, 1-5, 2-3, 2-4, and 2-5). The current upper bound is given by Joel Spencer et al, using more advanced graph theory[5] $U \leqslant O(N^{\frac{4}{3}})$, and the lower bound for the moment, derived from special lattice case, is[1] $U \geqslant O(N^{1+\frac{c}{1+\log\log N}})$.

# 2 Constructions Derived from Lattice Sections

One way to construct takes the advantages of periodic lattice. Suppose a Cartesian coordinate, we place at each integer coordinate a point, namely $(x, y)\ \forall x, y \in \mathbf{Z}$. Then every point has 4 neighbors, giving $U = 2N$ or $u = 2\rho$. Furthermore, in the case of a finer lattice, $|\mathbf{a}| = |\mathbf{b}| = \frac{1}{5}$, the connectivity of each point rises to 6. For example, to point at (0, 0), the 12 valid neighbors are $(\pm 1, 0)$, $(\frac{4}{5}, \pm\frac{3}{5})$, $(\frac{3}{5}, \pm\frac{4}{5})$, $(0, \pm 1)$, $(-\frac{4}{5}, \pm\frac{3}{5})$, $(-\frac{3}{5}, \pm\frac{4}{5})$, giving $U = 6N$ or $u(\rho = 25) = 6$. The construction above is shown in Fig [2]. Generally, if the side length of the cell is $|\mathbf{a}| = \frac{1}{\sqrt{m}}$, then the connectivity equals one half the number of *integer* solutions of $x^2 + y^2 = m$.

If in an arbitrary lattice, the basic vectors become $\mathbf{a} = (\frac{p}{\sqrt{m}}, 0), \mathbf{b} = (\frac{q}{\sqrt{m}}\cos\theta, \frac{q}{\sqrt{m}}\sin\theta)$, and there is more than one point in a primitive cell. But to calculate the asymptotic expression, only two points in the cell need to be taken into account. We can choose the coordinate so that one is $\mathbf{p}_1 = x\mathbf{a} + y\mathbf{b}$, the other is represented as $\mathbf{p}_2 = (\frac{x_0}{\sqrt{m}}, \frac{y_0}{\sqrt{m}}) + x\mathbf{a} + y\mathbf{b}$, where $x, y$ run over all integers, then the equation turns to be

$$(x_0 + xp + yq\cos\theta)^2 + (y_0 + yq\sin\theta)^2 = m \tag{1}$$

or

$$\begin{gathered} Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0 \\ A = p^2, \quad B = 2pq\cos\theta, \quad C = q^2, \quad D = 2px_0 \\ E = 2x_0 q\cos\theta + 2y_0 q\sin\theta, \quad F = x_0^2 + y_0^2 - m \end{gathered} \tag{2}$$

The number of integer solutions $(x, y) \in \mathbf{Z}^2$ of equation[1,2], denoted as $f(m)$, determines the connectivity of point (0, 0). In analogy to the square grid formulated by $x^2 + y^2 = m$, we can infer that as $m$ increases, $f(m)$ can also increase, whose scaling in terms of $m$ is crucial(more rigorously speaking, we need $F(m) = \max_{m' \leq m} f(m')$, but here we write them both as $f(m)$). A grid with parameters $m, p, q, \theta$ and $f(m)$, will supposedly possess the density $\rho \sim \frac{m}{pq\sin\theta}$ and connectivity $u \propto f(m)$. Hence it is a way to obtain good configurations by means of powerful number theory.

The following theorems solve the lattice situation step by step[1, 3, 4]

**Theorem 1.** *If the coefficients $A, B, C, D, E$ and $F$ of Eq[2] contain irrationals and cannot reduce to be all rational, then $f(m) = O(1)$.*

If the coefficients of Eq[2] have irrational numbers and the equation has sufficient integer solutions, we can express the irrational coefficient by solutions, thus it must be rational.

**Theorem 2.** *All quadratic forms $Ax^2+Bxy+Cy^2+Dx+Ey+F=0$ with all coefficients integer and $B^2-4AC \neq 0$, can be reduced to $Ax^2+Bxy+Cy^2+F'=0$ by changing variables linearly where $F'$ is linear to $F$ thus linear to $m$ in Eq[1].*

**Theorem 3.** *The number of integer solutions of equation $x^2+y^2=m$, where $m$ is an integer greater than 0, equals to $4|h_1(m)-h_3(m)|$ where $h_1(m)$ is the number of positive divisors of $m$ that take the form of $4k+1$, and $h_3(m)$ is the number of positive divisors that take the form of $4k+3$.*

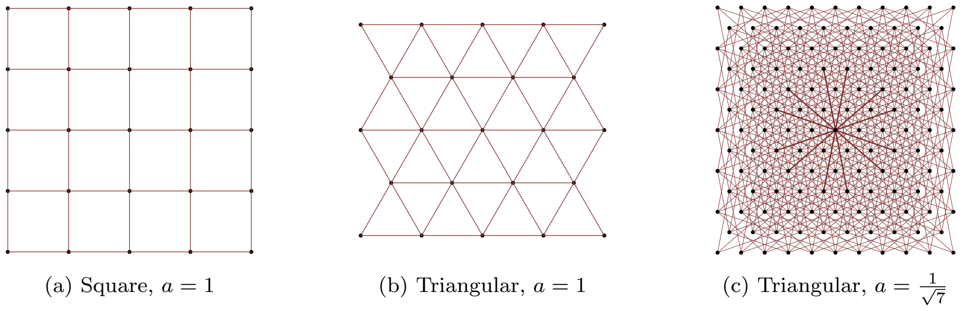
Take $m=25$ as an example: 25 has 3 positive divisors that are 1, 5, and 25. They all equal $1 (\mathrm{mod}\ 4)$. Hence $h_1(25)=3$ and $h_3(25)=0$, the equation $x^2+y^2=25$ has $4\times 3=12$ integer solutions.

**Theorem 4.** *Let $d(m)$ be the number of divisors of $m$, then $\overline{\lim}_{m\to\infty} \frac{\log d(m) \log\log m}{\log m} = c$ where $c$ is a constant. In other words, the upper limit of $d(m)$ for $m\to\infty$ scales as $m^{\frac{c}{\log\log m}}$.*

**Theorem 5** (Erdös 1937)**.** *In the case of $x^2+y^2=m$, $f(m)$ has the lower bound $O(m^{\frac{c}{\log\log m}})$.*

The theorem[1] restricts all the reasonable constructions of lattices to have points only at rational coordinates. The theorem[3], relates the number of solutions to factorization of $m$, though we for the moment only find conclusions for a few simple quadratic forms. Since neither $h_1(m)$ nor $h_3(m)$ can exceed $d(m)$, theorem[4] is essentially the upper bound of $f(m)$. Combining with theorem[5], we conclude that it is the definite expression describing square lattice. Other types of lattices, corresponding to other quadratic forms. For example, triangular lattice is formulated by $x^2+xy+y^2=m$ and $f(m)=6|g_1(m)-g_2(m)|$ in which $g_1$ and $g_2$ are numbers of divisors taking the forms of $3k+1$ and $3k+2$ respectively. The theorem[4] can still bind $f(m)$ yet theorem[5] has to be restated.

To make the reasoning complete, a consistent upper bound is needed for the number of solutions of all the quadratic forms, with respect to $\rho = \frac{m}{pq\sin\theta}$. Nevertheless, I will not delve into it in this report and regard the behavior $u \sim O(\rho^{\frac{c}{\log\log\rho}})$ with $c$ a constant to be suitable to all lattices.



(a) Square, $a = 1$

(b) Triangular, $a = 1$

(c) Triangular, $a = \frac{1}{\sqrt{7}}$

Figure 2: Examples of lattice solutions. (a) Square grid with side length of smallest square $a = 1$, $U(N) = 2N$. (b) Triangular grid with $a = 1$, $U(N) = 3N$. (c) Triangular grid with $a = \frac{1}{\sqrt{7}}$, $U(N) = 6N$.

# 3 Numerical Methods

The analysis of constructions derived from lattices to the unit distance problem does not establish proof of the optimality over all the graphs. Hence it is the numerical trials that become main focus of our work. A naive numerical approach is to enumerate all the possibilities of a graph containing $N$ vertices, and to check that they are embeddable into the plane. This algorithm by brute force renders has exponential temporal cost, thus we compute numerically mainly through randomized non-optimal algorithms which are Monte Carlo simulated annealing(MCSA) and greedy extension.

The following five classes of algorithms are used in order to obtain graphs of large number $N$. MCSA that generates various configurations with fewer points, serves as the basics of the whole framework. Greedy algorithms thereafter expand in some certain ways each configuration to that of greater number of points. Symmetry consideration is employed in MCSA, but can potentially be generalized in all kinds of algorithms. Densest subgraph is an algorithms to analyze the outcome of MCSA, extracting the optimal subset from a given configuration, especially suitable to the unit distance problem. Finally, a separate part is devoted to the algorithm that makes the outcome of MCSA an accurate one.

## 3.1 Monte Carlo Simulated Annealing

The first idea leading to MCSA is to shift the original problem from maximum number of unit distance pairs, to minimum energy of the system. Assume that all the $N$ points are able to move freely on the plane, interacting with each other with specific energy expression. We hope the ground state to be the one that consists of most pairs of unit distance. We thus design the pair potential as

$$E(\mathbf{r}_i, \mathbf{r}_j) = E(r = |\mathbf{r}_i - \mathbf{r}_j|) = \begin{cases} -E_0 * \frac{r-r_0}{r_0} & : \quad 0 \leqslant r \leqslant r_0 \\ -r^p & : \quad r_0 < r \leqslant 1 \\ -(2-r)^p & : \quad 1 < r < 2 \\ 0 & : \quad r \leqslant 2 \end{cases} \tag{3}$$

where $r_0 < 1$. $E_0$, $p$, and $r_0$ are adjustable parameters. As plotted in Fig [3], the potential mainly consists of two part: 1. a wedge-like repulsive core, to push the particles away from each other, to avoid the coincidence of them which would lead to a trivial $U(N) \sim N^2$ construction. 2. a sharp trap favoring particles to sit on the circle of radius 1.0. Oftentimes a hard core $E = E_\infty$ when

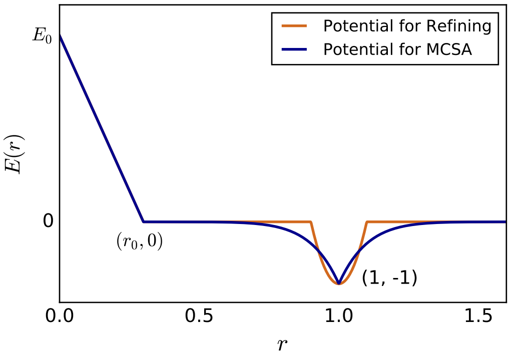


Figure 3: The sketch of pair potential used in MCSA and to refine outcomes. For MCSA, it consists of a repulsive core and a sharp convex trap centering $r = 1$. For refining, around $r = 1$ a concave expression is used instead.

$r < r_{hc}$ with $E_\infty$ sufficiently large and $r_{hc} \sim 0.01$ is added to guarantee non-overlapping.

Here I need to emphasize that the energy function[3] above is only one choice aimed to maximize the number of one-length edges. Not only the parameters inside can change, the form of the curves is flexible, even the 3-or-more-point interaction could be included. However, it is the simplest form that I use throughout the work and find it most efficient.

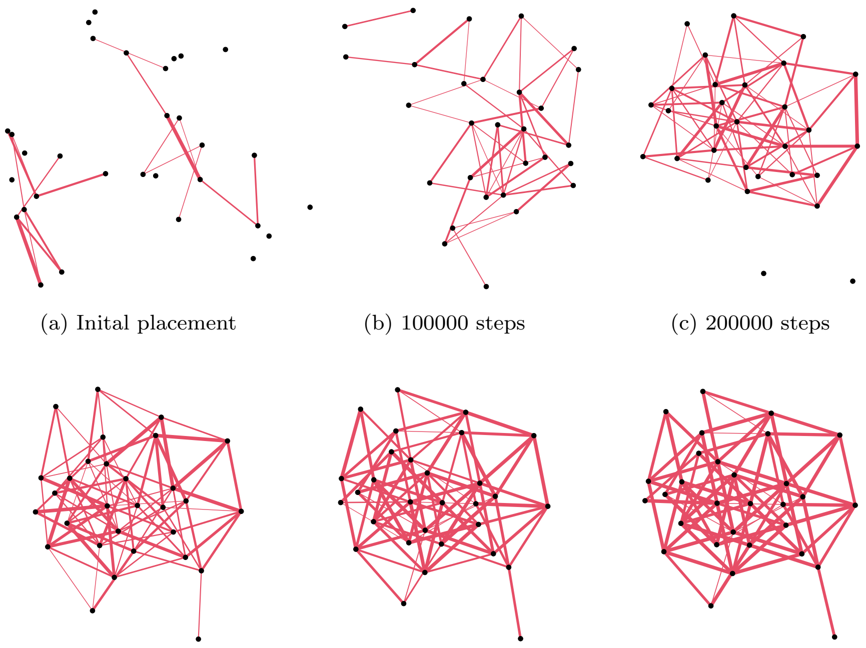
More remarks about the shape of the potential:

- Convex curve around $r = 1$ is essential to the criteria "pairs with distance exactly equal to 1", but leaves profound difficulties to refine the result. See the subsection[3.5].

- The width of repulsive and trapping parts of the potential need also careful consideration and tuning. The width of the core should be much greater than the trap's. We hope the trap to be wider to attract more particles on the one hand, but must keep a decent width of the core.

- Do not use $\frac{1}{|1-r|^p}$ because once a particle falls into a singularity it can never escape.

Given the potential, the next step is to find the minimum over all the possible configurations. The phase space is so huge that no one can find the exact minimum by searching. Monte Carlo simulated annealing is a random algorithm to deal with this problem in finite steps, though is still unable to assure the optimality. The idea is: each time we change the current configuration a little bit, to a random neighboring point in the phase space; calculate the energy change $\Delta E = E_{new} -$



(a) Inital placement

(b) 100000 steps

(c) 200000 steps

(d) 300000 steps

(e) 400000 steps

(f) Final position

Figure 4: T evolution of the configuration in a typical MCSA process, $N = 30$, $\beta_{inital} = 0.5$, $\beta_{final} = 20.0$. The thickness of lines between two points indicates how close their distance $d$ is around 1, thinnest ones for $|d - 1| = 0.1$ and thickest ones for $|d - 1| < 0.001$

$E_{old}$, the probability to accept the move equals to $\min(1, e^{-\beta \Delta E})$, otherwise remain unchanged. The parameter $\beta$, interpreted as reciprocal of temperature in physical context, should increase over steps, $\beta_{n+1} = 1.0001\beta_n$ for example.

It is readily to see when $\beta \to \infty$ the program is thoroughly seeking for the local minimum, whereas it can jump over barrier when $\beta$ is finite, which allow us to possibly locate the global minimum. In practice, I take each move as the displacement $[x - \delta, x + \delta] \times [y - \delta, y + \delta]$ where $(x, y)$ are original coordinate of the point of choice, and $\delta$ the probability is set uniform within that area. In fact, the displacement $\delta$ is also quite an crucial quantity that affects the performance, which should in principle make the accepting rate be 50% for best efficiency. Thus we change it as well over steps, but keep it not too small(for example, once an intended move is rejected and if $\delta > 0.01$, $\delta \to 0.999\delta$). Now there are more parameters to be adjusted, among which the total number of

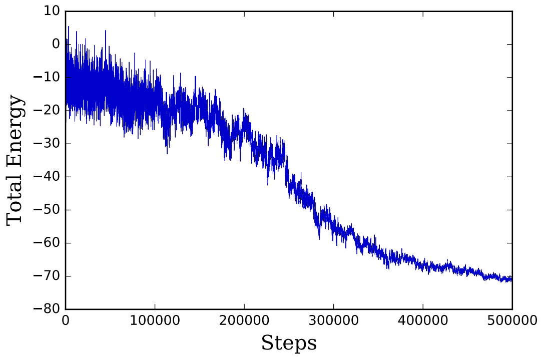


Figure 5: The evolution of total energy in the simulation of Fig [4].

steps $N_{step}$, initial $\beta_0$, final $\beta_1$ are most important and should rely on different numbers of points.

The evolution of total energy in Fig [5], only serves to test the performance of MCSA. As we try to minimize the total energy, it falls from the beginning random configuration, until some time followed by a plateau during which the energy almost remains constant, temperature low, and refining is in process. An effective MCSA needs to see this plateau but the main work of gaining accuracy belongs to the next procedure.

As for boundary condition, the simple hard-wall box is employed: if an attempted move $(x, y) \to (x + \delta x, y + \delta y)$ is taken, but $(x + \delta x, y + \delta y)$ is not in $[-L, L] \times [-L, L]$, then the move is always rejected. We have thought about the usage of periodic boundary condition(PBC) but finally gave up. In that case, points are actually moving on a torus and the solution is essentially lattice. However, more disadvantages than improvements may take place by adopting PBC:

- More parameters are introduced which are the side lengths of the periodic area, $L_x$ and $L_y$, plus its angle $\alpha$. We not only have to run programs for different $N$ but also need to determine which size lengths can give rise to maximal connectivity. In fact, a certain good outcome only appear in very certain and accurate $L_x$, $L_y$, and $\alpha$. A possible remedy is to anneal $L_x$, $L_y$ and $\alpha$ at the same time, which may lower the efficiency and bring in much programming work.

- The decisive aspect, of purely technical nature, which makes us give up PBC, is that all the following techniques, from densest subgraph to refining, must be rewritten for a PBC version. Symmetrical aspect is the most tedious, because there are 17 members in planar translational-invariant structures, called wall-paper groups. Each of them needs its own programming if we take all of them into account.

- As stated in section[1], finite number of points is equivalent to periodic cases, meaning that we do not lose any solutions in non-periodic condition, just a drawback in efficiency.
- The greedy algorithm described below, will provide an approach to lattice on top of finite points.

## 3.2 Densest Subgraph

To avoid ambiguities, let us define the densest subgraph used for the problem.

**Definition.** *A simple undirected unweighted graph $G$ is a set of vertices $V = \{V_1, V_2, \dots, V_N\}$ and a set of distinct edges $E = \{E_1, \dots, E_M\}$ where $E_i = \{V_{i1}, V_{i2}\}$ are unordered pairs of different vertices. A subset of $V$, denoted as $V'$, can induce a subset $E' = \{\{V_i, V_j\},\ \forall V_i, V_j \in V,\ and \{V_i, V_j\} \in E\}$.*

**Definition.** *The density of a graph is $\rho(G) = \frac{|E|}{|V|}$, and the densest subgraph is naturally the subgraph of $G$ with maximal density.*

Densest subgraph is enough to extract the optimal part from raw configurations, and it might potentially be generalized to a non-simple (self-loop or multiple edges for one vertex pair) weighted graph if one vertex can represent more than one point on the plane (for instance, to simulate the periodic situation, a vertex can be points repeating in each cell).

The algorithm of the densest subgraph is quite complicated, based on maximum-flow network, which can be found in ref[2]. It is too lengthy to describe here; instead, we will outline this algorithm as follows.

The concept of maximum flow is that, given a directed weighted graph (the weigh of each edge is also called *capacity* here), one vertex $S \in V$ is called *source*, and another vertex $T \in V$ is called *sink*. A *cut* is the subset of edges without which there is no path from $S$ to $T$, and the cut with the minimum sum of their capacities is the minimum-cut (here, we use the equivalence between the maximum flow and minimum cut). Each cut disconnects the graph; the part with $S$ is called the *$S$-cut*, and the rest is *$T$-cut*.

Back to the densest subgraph, we rewrite the problem $\max_{G' \in G} \frac{|E(G')|}{|V(G')|} \Rightarrow \max_{G' \in G} |E(G')| - \lambda |V(G')|$. For a certain $\lambda$, if the right-hand problem has a positive answer, the answer of the left-hand problem must be at least $\lambda$; otherwise, the answer of the left-hand problem must be less than $\lambda$. A binary-search scheme is used to find the proper $\lambda$ resulting in a zero answer for the right form, which is the exact maximum density of the raw graph.

As for the subtracting version of problem, we construct a new graph by adding a dumb source $S$ and a dumb sink $T$ to the original graph. The capacities in the new graph are set to be: 1. If $v_1$ and $v_2$ are two nodes in the original graph with an edge in between, then $c(v_1, v_2) = 1$; 2. for any node $v$ in the original graph, $c(S, v) = M$, and $c(v, T) = M + 2\lambda - d_v$, where $M = |E|$ is the number of edges in the original graph, and $d_v$ is the degree (the number of edges connected) of node $v$. After the new graph is constructed, calculate its maximum flow, as well as its $S$-cut. The answer, namely maximum value of $|E(G')| - \lambda|V(G')|$, is just $maxflow - M \times N$, and the $S$-cut is the corresponding subgraph. See[2] for the proof of their equivalence.

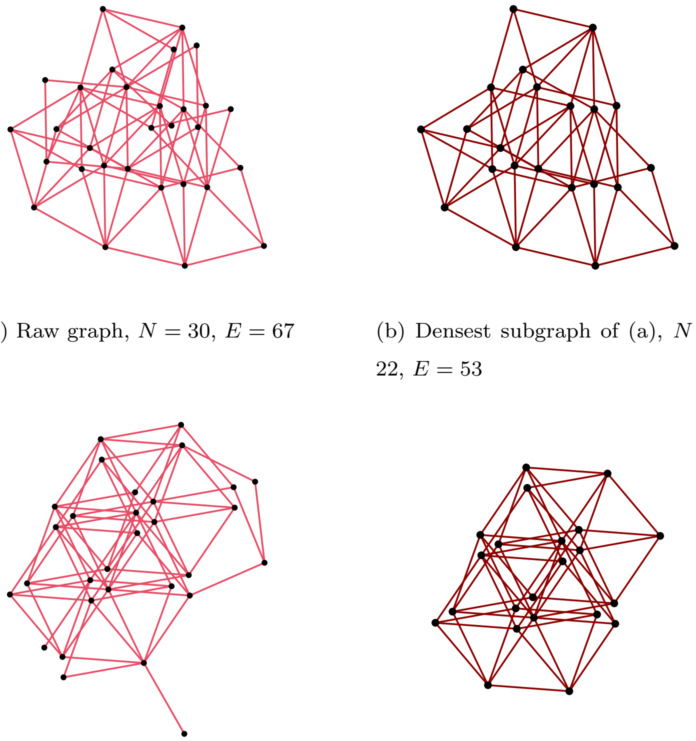
The densest subgraph is used to extract the densest subsets of MCSA's results. Vertices are points, and edges are pairs of points of distance 1, (plus a small tolerance). Its idea can simply be stated as "more edges with fewer points," but we finally find it of great significance to the unit distance problem because the connectivity is, in some sense, more essential than the pure number of edges. Put rigorously, since $U(N) \sim N$ relation is concave, or $U''(N) > 0$, when a subgraph of a higher density $\frac{|E(G')|}{|V(G')|} > \frac{|E(G)|}{|V(G)|}$ is found, $G$ can no longer be an optimal graph. As shown in Fig [6], this algorithm removes the points that are isolated or with few links.

### 3.3 Symmetrization

It is worth re-emphasizing that there is still no proof that optimal constructions bear any symmetries (rotational or translational). However, on the one hand, we frequently observe very good graphs possessing very beautiful symmetries, and on the other hand, including symmetries can produce the results with a greater number of $N$.

For finite $N$, we consider only rotational symmetries, represented by planar point groups (translational symmetries need wallpaper groups that are too complex to program). Luckily, only two types of transformation are involved: 1. $C_n$, $n$-fold rotations and 2. $D_n$, $n$-fold rotations plus $n$ mirror axis. Examples are the results in Fig [10].

For each symmetry, we need only one piece of the plane to represent the whole, and points are duplicated from that piece. $\frac{1}{n}$ of the whole plane is required (sector of $\frac{2\pi}{n}$) in $C_n$, yet only $\frac{1}{2n}$ of the plane is needed for $D_n$. The central point in $C_n$, and points that are central or along the side of



(a) Raw graph, $N = 30$, $E = 67$

(b) Densest subgraph of (a), $N = 22$, $E = 53$

(c) Raw graph, $N = 30$, $E = 71$

(d) Densest subgraph of (c), $N = 22$, $E = 56$

Figure 6: Examples of densest subgraph. (a) and (c) are raw results directly given by MCSA, and (b) and (d) are, respectively, their densest subgraphs, namely the subsets maximizing $\frac{|E|}{|N|}$.

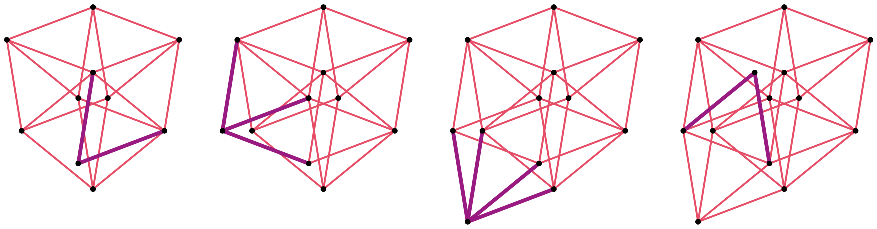


Figure 7: 4 consecutive single-point addition to the configuration $N = 9, E = 18$ in Fig [1f]. Thick purple lines and the extra points represent the new points and links involved each time.

the sector chosen in $D_n$, are treated differently in the manner of displacement, and in counting the total energy. All of the above will definitely complicate the coding, but it is worth a try in pursuit of the nice results.

## 3.4 Greedy Expansion

Greedy algorithm is, in general, a strategy to make locally optimal choices at each stage in order to achieve a global optimum. They are not difficult to devise, for our unit distance problem, but they are quite powerful to generate graphs with great numbers of points. Starting from two points of distance 1, we can add points one by one, each time maximizing the number of links, namely to place the new point at the coordinate that has the most 1-distanced neighbors; then what we arrive at is trivially nothing but a triangular lattice. However, if we start from the existing solutions of MCSA, the results are much more interesting and promising. Here are two types of greedy algorithms that have proven effective for the problem.

### Single-Point Addition

As described above, the most straightforward method is to add points in a greedy manner, optimizing the number of links when placing each point. Fig [7] shows the process of the algorithm working on graph 9-18 (we usually in this report denote a graph as $N$-$E$). To make our configuration look compact, we place the point as closest as possible to the origin, namely the center of the configuration, where exist many optimal new points.

A technique-related question arises—how to find the best position for placing the new point, given a configuration? Obviously, this best point must bring in at least two new links; thus, we can enumerate all of the pairs, find the coordinates with equal distance between them (they are just two vertices of a rhombus), and count how many new links are with these new places. Then select the position with maximum links added, and closest to the origin. This scheme by brute force costs $O(N^4)$ of time (to add $N$ points, each time the program scans $O(N^2)$ pairs, and for each possible place, counting the new links costs $O(N)$).

A more efficient algorithm is: If one position has more than one unit-distanced neighbor and is not occupied by any points, we call it a virtual point, and keep track of all of them as well as the number of their potential links. We maintain a *heap* to select the new optimal point, and at the same time, we divide the whole plane into small grids, and we record the virtual points in each cell to efficiently trace a virtual point by location. Once a virtual point is selected to place the new point, it is deleted, some new virtual points are added, and some are updated (their numbers of links increase by 1). The time cost of this scheme is $O(N^2 \log N)$. In practice, it takes only a few seconds to add 5000 points into an MCSA result.

**Duplication**

Another way to reach large-$N$ solutions is to duplicate the existing ones over and over again. The number of points doubles as $N \to 2N \to 4N \to \cdots$ (for the case $N \to 2N \to 3N \to \cdots$, it proves to be less optimal). Duplicating a graph with $N$ points and $U$ edges can bring in at least $N+1$ edges, just by making corresponding points connected, which could be written as $U(2N) > 2U(N) + N$. Excitingly, this naive mean has already led to $U(N) \sim N \log N$ according to the master theorem. A better arrangement of the two raw configurations can hopefully result in excellent combinations.

The arbitrary relative placement of two identical configurations has too many possibilities beyond the capacity of our computer, and constraints must be employed to reduce the searching space. Different constraints render various versions of duplications. By testing or intuition, several constraints are considerable for optimal arrangements:

1. Do not rotate any of the two raw configurations; just translate one of them.

2. If overlapping is not allowed, at least one point from one side connects more than one points of the other side. In fact, we can envision that there must be at least two points from one side, each connecting more than one point from the other side.

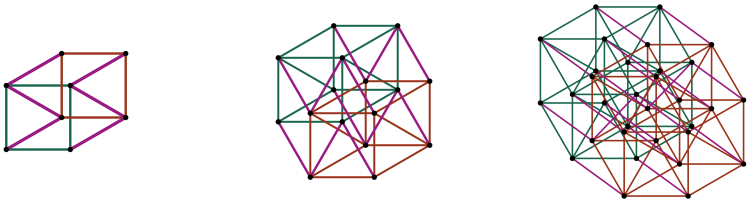


Figure 8: Example of duplication. Start with a square, and non-overlapping greedy duplication is employed. Dark green and brown edges come from two initial configurations, and purple edges are newly generated in between. The number of points increases as 4 → 8 → 16 → 32, yet edges are 4 → 14 → 40 → 98.

3. If overlapping is allowed, we can enumerate the two points on both sides that overlap. We assume that two pairs of overlapping points can accelerate, which is almost always true.

In practice, overlapping or non-overlapping versions of duplication are simultaneously adopted, varying according to different raw configurations. Fig [8] shows the process of the duplication of a square.

## 3.5 Refining the Results

Accuracy issue arises from MCSA due to the "single-point displacement" scheme bearing metastable states, and to the finite number of steps. Look at Fig [9]; particles 1 and 4 are locally stable. For 2 and 3, it suffices to study the movement along the link 2-3, because the movement(of particle 2, for example) perpendicular to the edge 2-3 will destroy the two existing unit links 1-2 and 2-4. Particle 2 moving $\delta$ upward elongates link 2-3 by $\delta$, and it also extends 2-1 and 2-4 each by $\frac{\delta}{2}$. Since the potential around $r = 1$ is convex, this displacement will eventually increase the total energy. A downward displacement of particle 2 is unstable since it shorten all of the three links.

To improve the accuracy to the limit of the float-point truncation in computer, a makeshift energy expression is imposed that is concave around $r = 1 (E'' > 0)$ in place of the old convex one, shown in Fig [3]. However, a new challenge arises because if an edge has a distance very close to 1.0 but does not belong to the final valid links, this may destroy the entire solution. Hence, only a

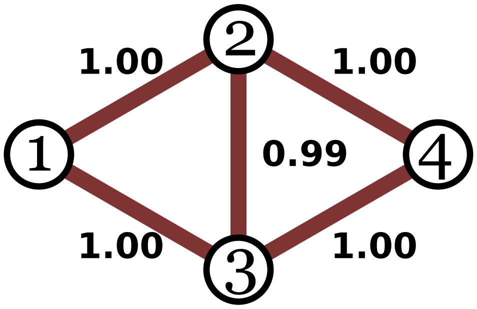


Figure 9: A metastable state with respect to any displacement of a single point, using the energy function convex around 1.0. This is a counter-example leading to inaccuracy of MCSA.

“partial” refining scheme works: Select from an MCSA result the edges in the range of $(1-\epsilon, 1+\epsilon)$, and minimize the total energy of the new form concerning *only* the set of the edges already chosen. The parameter $\epsilon$ is also experiential, usually 0.01.

Now, the minimization no longer requires annealing, for we just seek the local minimum. We use a first-in-first-out queue to stabilize the particles sequentially. The whole process is:

1. Choose a point with at least one links that deviates from 1, and push it into the queue.

2. Pop the first element from the queue, and find the local minimum around it using the directional partial derivative and golden-ratio searching scheme. Keep in mind that we only take certain selected links into account and a concave function potential is working.

3. If the displacement of that point after minimization is beyond $err$, the accuracy we want, then check all of its neighbors. Push those who are not in the queue into the queue.

4. Go to step 2 until the queue is empty.

## 3.6 the Process of the Whole Program

Below, we list all the algorithms used in this project:

1. Start with a certain number of points randomly distributed in the plane, or in a sector according to the symmetry imposed.

2. Annealing. Choose an arbitrary point, displace it in the range of $\delta$, accept the move by probability $\min(1, e^{-\beta\Delta E})$, and raise $\beta$ step by step.

3. Take those edges with lengths within $1 \pm \epsilon$, for the steps of densest subgraph and refining.

4. Calculate the densest subgraph with respect to chosen edges, remove sparse vertices and edges.

5. Refine the result to make the accuracy close to float-point truncation error.

6. Extend it farther by greedy algorithms.

Steps 4 and 5 are usually exchangeable in practice, and the greedy algorithms are regarded as complement part based on former outcomes.

# 4 Results and Discussions

## Parameters

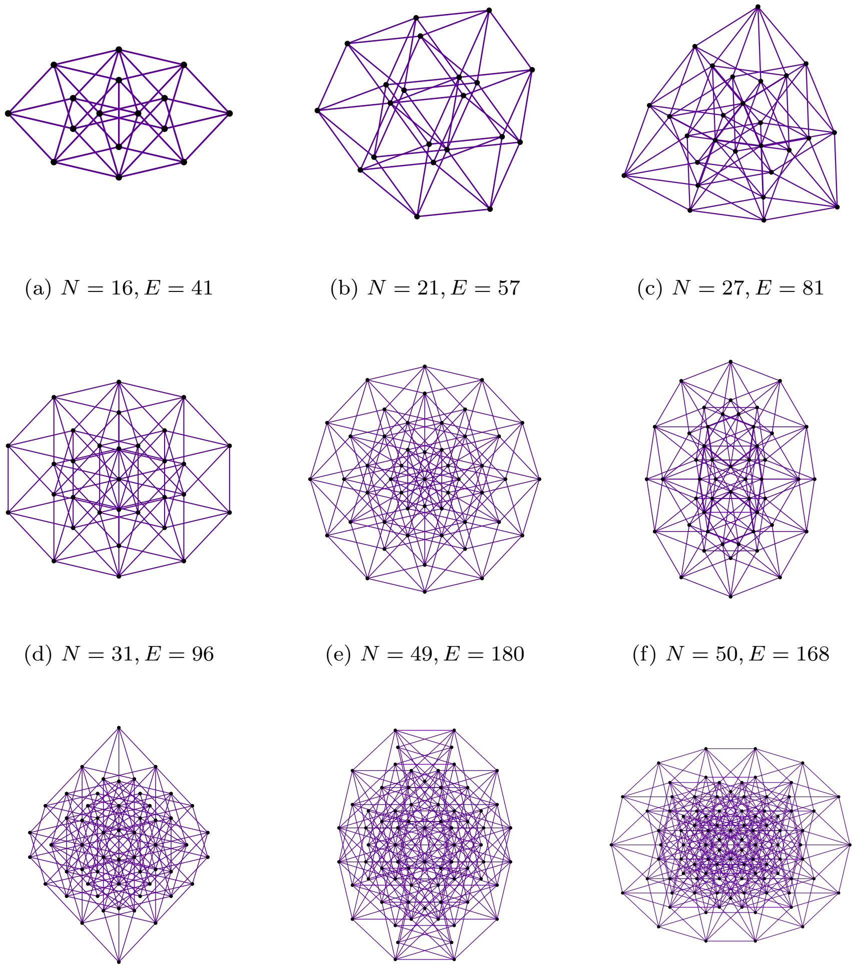
Systems: laptop PC, Windows 10, C++
Parameters: Take $N = 15$ and $D_2$ symmetry (thus 60 points in reality) as an example. Here are the optimal parameters: 1000000 steps of annealing in total, $\beta_0 = 0.5$, $\beta_1 = 1000$, $\epsilon = 0.01$, binding box $R = 3.0$, final deviation $err < 10^{-14}$; as for the potential, $E_0 = 8$, $r_0 = 0.25$, $p = 24$; initial range of trial displacement $\delta = 1.0$; in each 20 steps of annealing, when the accepting rate is less than 0.3, we set $\delta \to \delta/2$.
Cost of time: Also for $D_2$ and $N = 15$, about 5 seconds are needed to generate one configuration, most of which is devoted to annealing. The densest subgraph almost takes no time, and refining takes no more than half a second. For greedy expansion, single-point addition is usually as fast as a few seconds to treat one input, but duplication can only solve $N$ up to 1000 in several minutes.

Triangular grids are used for the purpose of comparison below and are constructed in a naive way: Place the first point, then the nearest 6 points clockwise, then the second nearest 12 points, then the next 18.... Only one parameter, the distance between two adjacent points $a$, determines the grid. We abbreviate the four kinds of triangular grids, of $a = 1$ as *TG1*, $a = \frac{1}{\sqrt{7}}$ as *TG2*, $a = \frac{1}{7}$ as *TG3*, and $a = \frac{1}{\sqrt{91}}$ as *TG4*. As analyzed before, these lattices would have, respectively, $U = 3N$, $U = 6N$, $U = 9N$ and $U = 12N$ as $N \to \infty$.

## 4.1 MCSA, before Expansion

Several representative samples with much aesthetics are shown in Fig [10], yet more results are included in the appendix. In this phase, we have not employed greedy algorithms, but the symmetrization and densest subgraph are included. Our current records, with different numbers of points, are plotted in Fig [11].

At first glimpse, what amazes us is that in the range of $N \leqslant 110$ our records cling very close to the curve of *TG2*, which may be the best lattice section constructions that I find in this range. However, when $N > 110$ our program can never achieve the connectivity of lattice cases. One of the important features of these configurations is their rotational symmetries: More than half have $D_2$ symmetry (sample 16-41, 31-96, 50-168, 56-200, 72-275 and 96-307), some with $C_3$ or $D_3$ symmetry



(a) $N = 16, E = 41$

(b) $N = 21, E = 57$

(c) $N = 27, E = 81$

(d) $N = 31, E = 96$

(e) $N = 49, E = 180$

(f) $N = 50, E = 168$

(g) $N = 56, E = 200$

(h) $N = 72, E = 275$

(i) $N = 96, E = 397$

Figure 10: Selected outcomes of MCSA, in which $N$ is the number of points and $E$ is the number of pairs of unit distance. Densest subgraph and symmetrization are used. More results are included in the appendix.

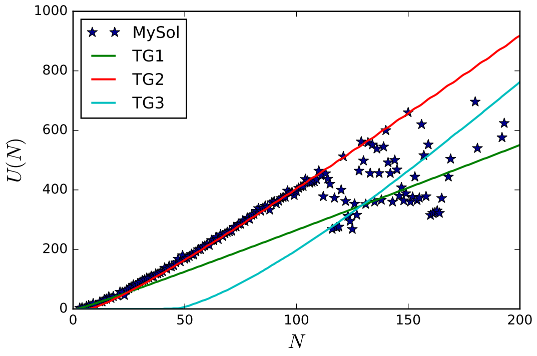


Figure 11: Our records of MCSA comparing with triangular grids. Horizontal axis is different number of points, and the vertical is number of links. Our solutions represented by dark blue stars are very close to triangular grid(*TG2*).

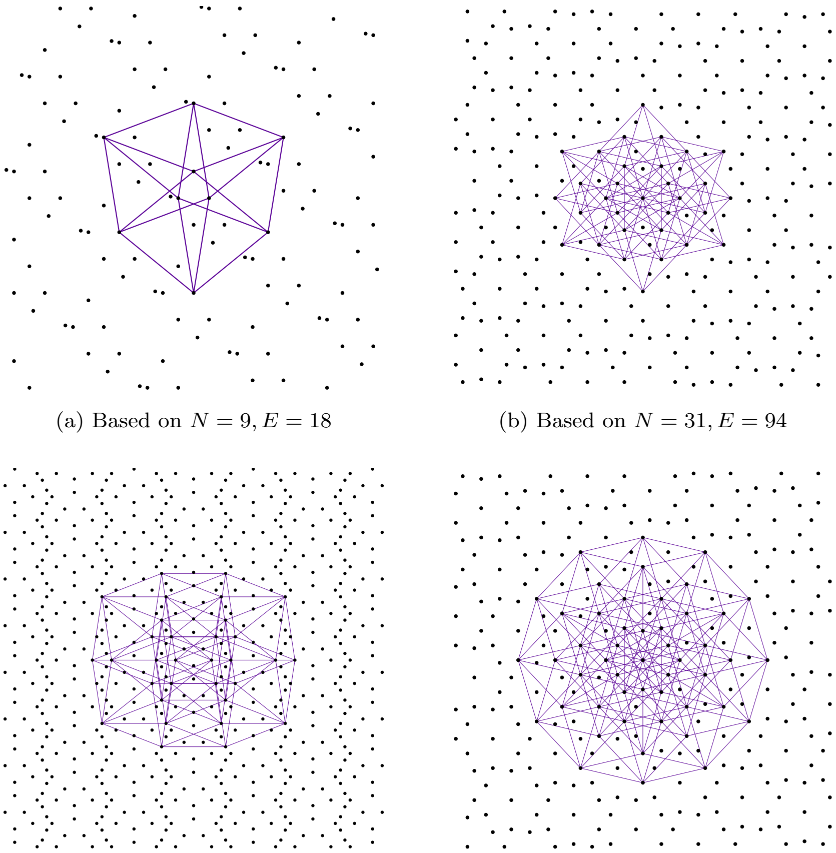
(sample 21-57 and 27-81), and a few with $D_6$ (sample 49-180). When we impose other invariances, such as $C_5$ or $C_7$, the results are obviously quite inferior, which may indicate that good solutions usually have simple lattices as their backgrounds.

We try but fail to generalize the results of MCSA to a universal approach to design solutions for an arbitrary number of points. In my opinion, each of the figures in Fig [10] has special regularity, and we can learn from their pattern to design excellent graphs of larger number $N$ that may potentially break the current lower bound. Taking graph 49-180 in Fig [12d] as an example, which shows up as a small peak in our record in Fig [11], one can place points from the center, circle-by-circle away, keeping a high connectivity.

## 4.2 Greedy Expansion

### Single-Point Addition

First, focus on the single-point version of greedy expansion. Take the graph 34-104, the one with greatest asymptotic slope among all graphs we have tested, as the starter; the number of links versus the number of points is plotted in Fig [13a]. Comparing it with triangular grids, we find it defeats *TG2*, though only a little, within the range $N < 500$, but is eventually surpassed by *TG3* and *TG4*. The slope remains unchanged in the large limit of $N$, too, which is less than the slopes of *TG3* and *TG4*.



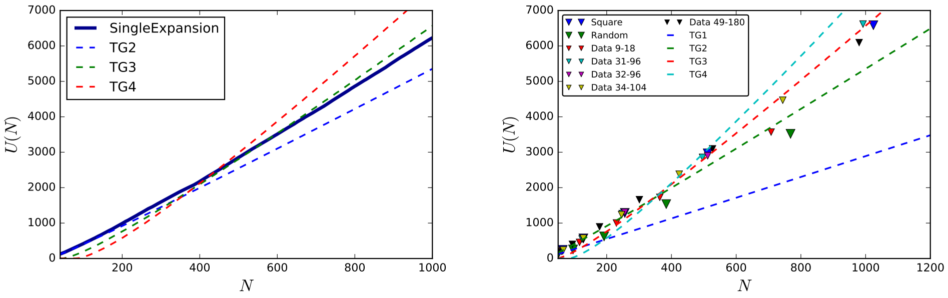
(a) Based on $N = 9, E = 18$

(b) Based on $N = 31, E = 94$

(c) Based on $N = 32, E = 96$

(d) Based on $N = 49, E = 180$

Figure 12: Selection of results of single-point greedy expansion, after 5000 points are added. The figures are only the sections of the whole 5000-point graph, and contain the raw constructions. See more in the appendix.



(a) Record of single-point addition of 34-104

(b) Record of duplications

Figure 13: Results of greedy expansions vs triangular grids. (a) Adding points greedily to graph 34-104. (b) The various results of the duplication algorithm, based on existing MCSA outcomes, except for one on unit-side-length square, and another on 6 random points.

More interesting are the expansion patterns for the large $N$ limit. The constant slope of $E - N$ relation of single-point addition in Fig [13a] implies the appearance of a periodic structure. In Fig [12] we look into for configurations after 5000 points are added. Almost all of the embeddings are periodic, and exceptions are displayed in Fig [14f] and Fig [14h] in the appendix. Outcomes with $C_5$ symmetry cannot usually expand into the lattice, as shown in Fig [14h]. It interests us that the lattices as the backgrounds, seem quite different from the raw configurations, and the primitive cells look very simple in contrast to the raw ones. For example, graph 49-180 has 49 points, but its background lattice has only 7 points in its cell.

Does it make sense if I say "good graphs are usually a part of the lattice"? In fact, any placement of finite points can be embedded in a periodic structure just by repeating itself, but here, the cells are much simpler than the raw configurations. If one can prove that the optimal configuration of $N$ points, constrained by $U(N)$ unit edges (now we already have the lower bound $U(N) \sim N^{1+\frac{c}{\log \log N}}$), must belong to the lattice whose smallest cell has $O(1)$ points, then the lattice case can prove to be the upper bound, and Erdös' problem will be solved.

### Duplication

Now turn to the duplication version of greedy algorithms. Here, we duplicate the solutions 9-18, 31-96, 32-96, 34-104, and 49-180 as well as a square and one configuration consisting of 6 random points. Their $U(N) \sim N$ relations are shown in Fig [13b]. Again, just like the record of single-point version, the outcomes of greedy duplication exhibit a slight advantage over triangular lattices in the range $N \leqslant 500$, and become inferior afterwards. However, Lacking regularities in the resulting configurations, we find it challenging to analyze and predict the behavior of further duplications.

In both versions of greedy algorithms, the densest subgraph and symmetrization are not used because: 1.The outcomes of expansion themselves are almost the densest sets and 2.algorithms without symmetries generally include symmetrical results.

## 5 Summary

I develop a series of algorithms combining Monte Carlo simulated annealing, densest subgraph, greedy expansion, and the consideration of symmetries, to approach configurations of as many unit-distance edges as possible. MCSA+densest subgraph+symmetrization can produce excellent configurations for $N \leqslant 110$, but so far they are still no better than triangular grids.

Greedy algorithm, by both single-point addition version and duplication version, can on the one hand generate considerable graphs that can surpass constructions of triangular lattices in the range $N \leqslant 500$, yet are defeated for larger $N$. On the other hand, for most of the outcomes of MCSA, periodic structures appear after adding sufficient points.

In terms of the record of $U(N)$ that I attain, no improvement can be made on the bound of $U(N)$. Nevertheless, several facts may imply that the constructions of lattices are optimal, because my records of MCSA only contain the graphs of symmetries compatible with lattices, and most of these graphs can be embedded into lattices of cells of points much fewer than raw configurations themselves.

Regarding the individual figures output from MCSA, they are each uique and very regular. I try but fail to generalize their rules to graphs for larger $N$.

## 6 Acknowledgment

I firstly thank Martin Zwierlein, at the department of physics in MIT, who contributes the original idea using MCSA to solve unit distance problem. And I acknowledge Tommaso Comparin for the discussion of details on implementation of MCSA and Jialiang Zhou for his profession in number

theory.

## References

[1] Paul Erdös. On the sum and difference of squares of primes. *Journal of the London Mathematical Society*, 1(2):133–136, 1937.

[2] Andrew V Goldberg. *Finding a maximum density subgraph*. University of California Berkeley, CA, 1984.

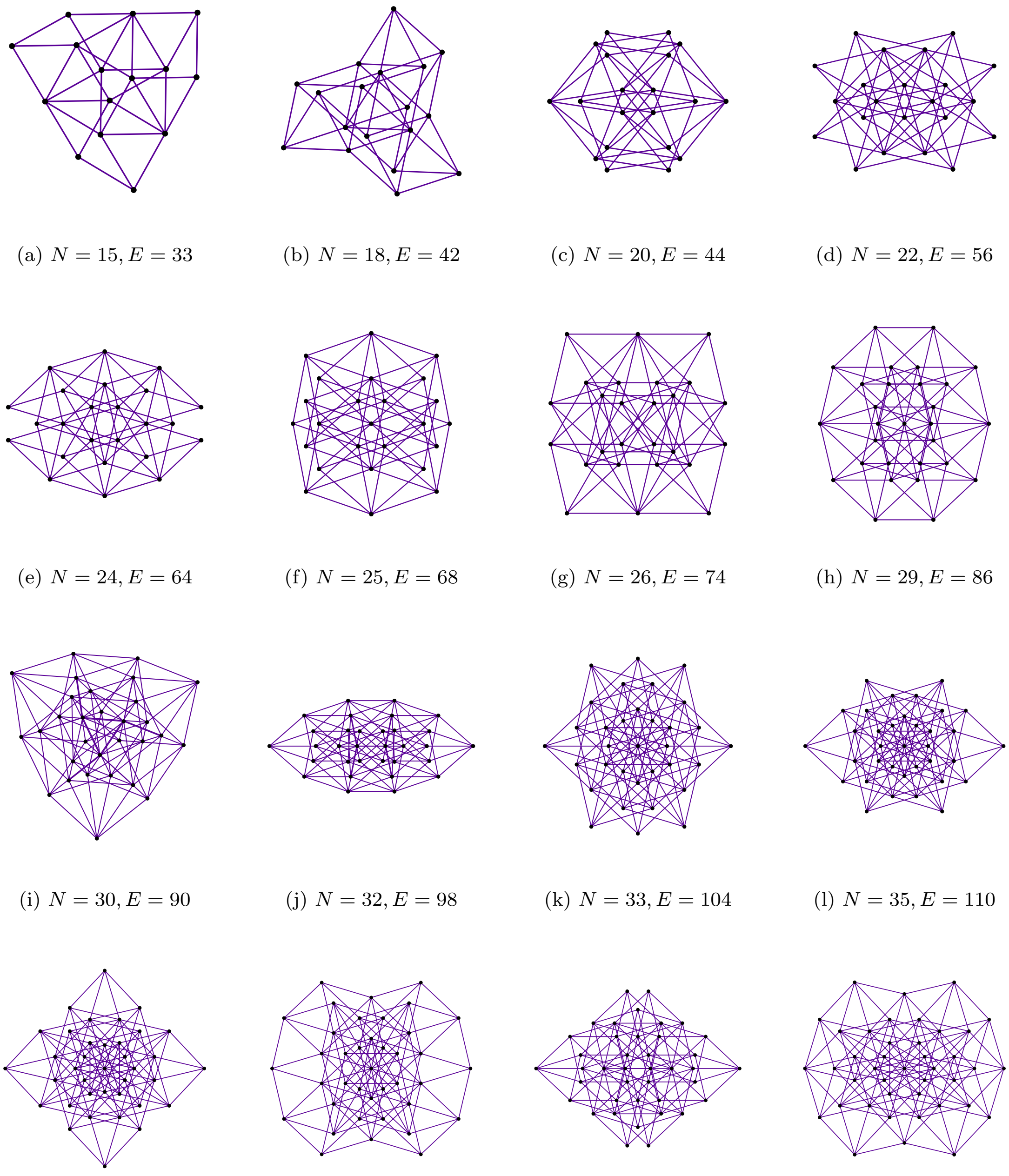
[3] Godfrey Harold Hardy and Edward Maitland Wright. *An introduction to the theory of numbers*. Oxford University Press, 1979.

[4] Luogeng Hua. *Introductions to Number Theory*. Beijing Scientific Press, 1957.

[5] Joel Spencer, Endre Szemerédi, and William T Trotter. Unit distances in the euclidean plane. *Graph theory and combinatorics*, pages 293–303, 1984.

# Appendix

More figures of MCSA and single-point greedy addition.



(a) $N = 15, E = 33$

(b) $N = 18, E = 42$

(c) $N = 20, E = 44$

(d) $N = 22, E = 56$

(e) $N = 24, E = 64$

(f) $N = 25, E = 68$

(g) $N = 26, E = 74$

(h) $N = 29, E = 86$

(i) $N = 30, E = 90$

(j) $N = 32, E = 98$

(k) $N = 33, E = 104$

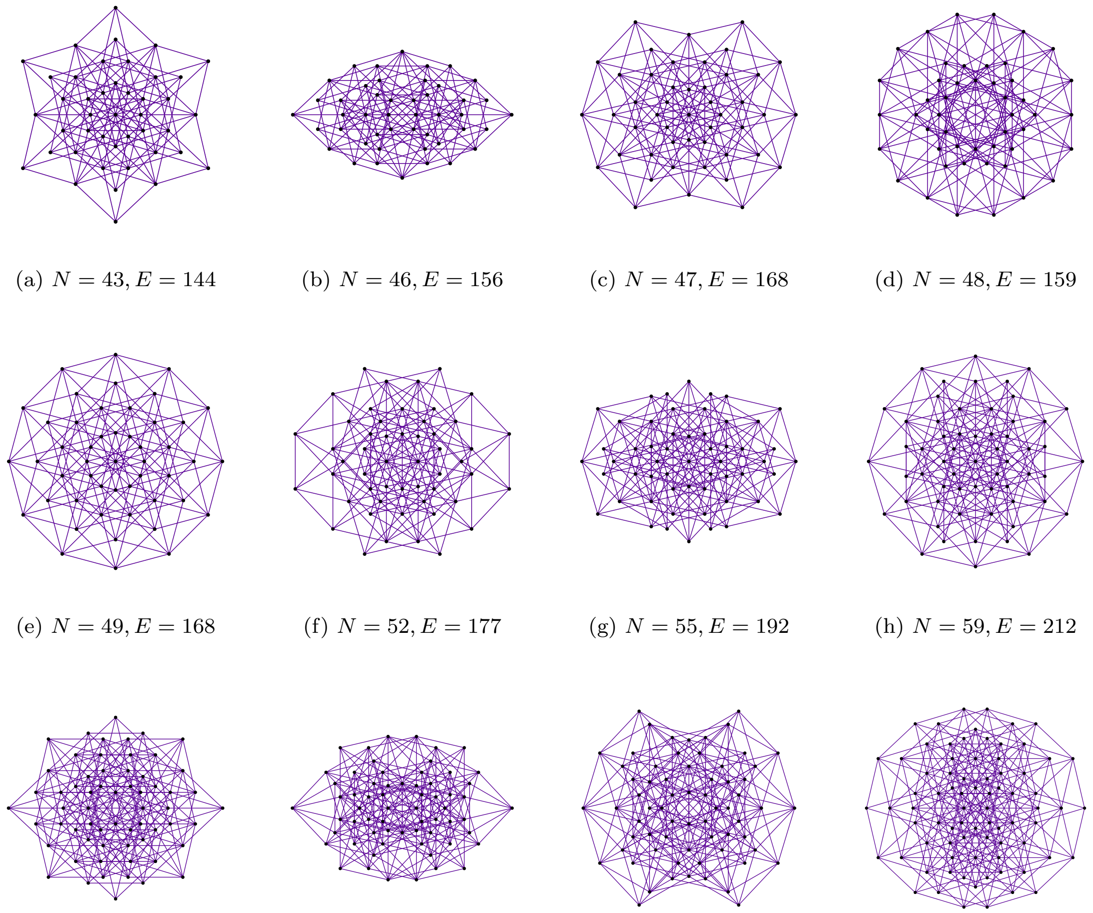
(l) $N = 35, E = 110$

(m) $N = 37, E = 118$

(n) $N = 39, E = 122$

(o) $N = 40, E = 126$

(p) $N = 41, E = 138$



(a) $N = 43, E = 144$

(b) $N = 46, E = 156$

(c) $N = 47, E = 168$

(d) $N = 48, E = 159$

(e) $N = 49, E = 168$

(f) $N = 52, E = 177$

(g) $N = 55, E = 192$

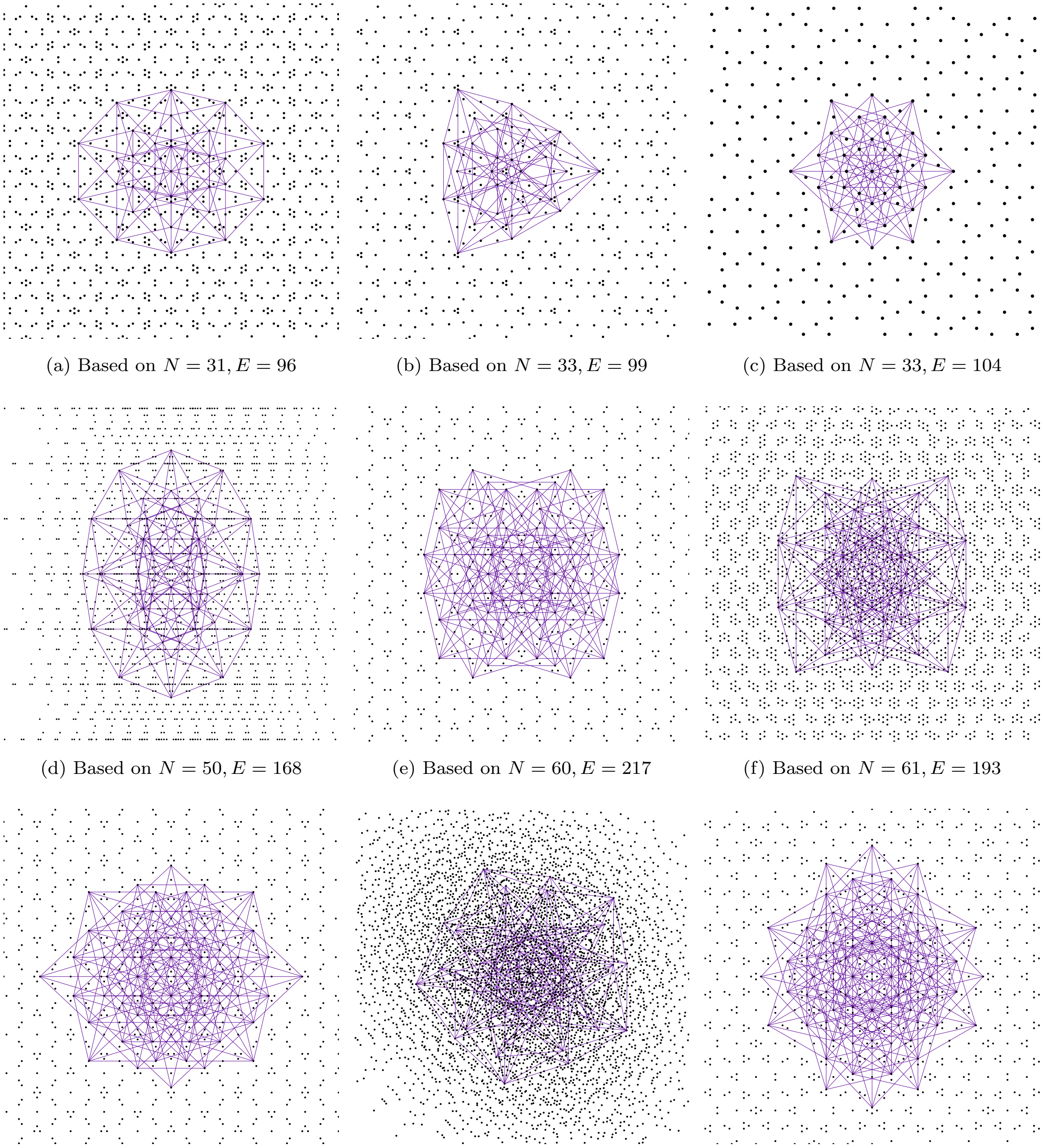
(h) $N = 59, E = 212$

(i) $N = 62, E = 231$

(j) $N = 64, E = 241$

(k) $N = 66, E = 250$

(l) $N = 78, E = 306$



(a) Based on $N = 31, E = 96$

(b) Based on $N = 33, E = 99$

(c) Based on $N = 33, E = 104$

(d) Based on $N = 50, E = 168$

(e) Based on $N = 60, E = 217$

(f) Based on $N = 61, E = 193$

(g) Based on $N = 62, E = 231$

(h) Based on $N = 65, E = 205$

(i) Based on $N = 72, E = 273$