Discrete Optimal Global Convergence of an Evolutionary Algorithm for Clusters under the Potential of Lennard Jones

Carlos Barrón-Romero
cbarron@correo.azc.uam.mx

Universidad Autónoma Metropolitana, Unidad Azcapotzalco
Av. San Pablo No. 180, Col. Reynosa Tamaulipas, C.P. 02200,
MEXICO

2016

Abstract

A review of the properties that bond the particles under Lennard Jones Potential allow to states properties and conditions for building evolutive algorithms using the CB lattice with other different lattices. The new lattice is called CB lattice and it is based on small cubes, such the number of its vertices in a region is always greater than the number of the particles of a cluster or a region of a lattice inside of the same size region of the CB lattice. Moreover, the estimation of a putative optimal cluster of the Lennard Jones can be done theoretically in short time but, the proof, for such cluster to be the global optimal cannot be determining in efficient time. The proof of the global optimality for a cluster is related to the binomial coefficient \( \binom{m}{n} \), which it corresponds with the selection of \( n \) particles from a collection with \( m \) given particles. A set of propositions states convergence and optimal conditions over the CB lattice for an evolutionary algorithm. The evolutionary algorithm is a reload version of previous genetic algorithms based in phenotypes. The novelty using CB lattice, together with the other lattices, and ad-hoc cluster segmentation and enumeration, is to allow the combination of genotype (DNA coding for cluster using their particle’s number) and phenotype (geometrical shapes using particle’s coordinates in 3D). A parallel version of an evolutionary algorithm for determining the global optimality is depicted. The algorithm for determining global optimality (which it is far from this research, and it is not included) is just a force brute searching algorithm with complexity \( \binom{m}{n} \), where \( n \) is the number of the cluster’s particles and \( m \gg n \) is the number of particles of an appropriate CB lattice’s region. The results presented are from a standalone program for a personal computer of the evolutionary algorithm, which can estimate all putative Optimal Lennard Jones Clusters from 13 to 1612 particles. The novelty are the theoretical results for the evolutionary algorithm’s efficiency, the strategies with phenotype or genotype, and the classification of the clusters based in an ad-hoc geometric algorithm for segmenting a cluster into its nucleus and layers. Also, the standalone program is not only capable to replicate the optimal Lennard Jones clusters in The Cambridge Cluster Database (CCD), but to find new ones.

Keywords: 02.60.Pn Numerical optimization, 21.60.Gx Cluster models, 31.15.Qg Molecular dynamics and other numerical methods, 36.40.Qv Stability and fragmentation of clusters, Lennard Jones Potential.

1 Introduction

The problem for determining optimal clusters under Lennard Jones captures my attention for the possible implications for building an efficient algorithm for the class of NP. My techniques for the NP Class has an application for building an appropriate algorithms for looking the optimal clusters under Lennard Jones Potential.

Over a decade ago, I states the conjecture in \cite{2} that IF lattice could contain all optimal clusters under the Potential of Lennard Jones (LJ). The title of the article: Minimum search space and efficient methods for structural cluster optimization was proposed as result of some inquiries from D. J. Wales, J. P. K. Doye, G.L. Xue and Bern Hardke about the optimal LJ clusters.
IF lattice results from overlapping the positions of the IC lattice and FC lattice. The main result was a minimum region of IF, where all putative global optimal LJ clusters from 2 to 1000 can be found (see figures 1 and 2).

Figure 1 was constructed by the selection of not repeated positions from where a given initial selection of particles converges always by a local minimization process to its putative minimal LJ cluster. This set of positions is finite, and it can be enumerate, such each position corresponds with a unique id number.

Therefore, it could be simple to locate a minimal LJ cluster by the set of its particles’ number of the cluster. I proposed a telephone algorithm, which is like make a phone call but, here $n$ id particles corresponds to a cluster’s phone number from a set of an appropriate selection of $m$ id numbers from a region of IF lattice. After the minimization if the value of the LJ potential is less than a previous cluster’s number, then it is the phone number of the cluster of $n$ particles. Even, it is like more genotype, I did not introduce this type of DNA mechanics in my previous genetic and evolutive algorithms in order to keep a phenotype representation (this means geometric shapes using the 3d particles’ coordinate). The main idea was to look for the putative optimal LJ cluster by an exhaustive searching. This is a brute force algorithm with complexity related to the Newton binomio for combinations, $\binom{m}{n}$.

Even with all putative optimal LJ clusters from 2 to 1000 in the lattice IF, I can not prove my conjecture. But, reviewing my previous work and the mathematical properties of the LJ potential function, it is possible to determine from the cubic lattice (CB) all optimal LJ clusters in efficient time.

This paper presents an evolutive algorithm based in our previous Genetic algorithm. It is based on the partial growing sequence property that the optimal LJ clusters exhibit (To my knowledge, it was Northby [12] the pioneer

Figure 1: MIF1739 contains the initial particles' positions for the $C_n^*, n = 2, \ldots, 1000$.

Figure 2: a) $C_{38}^*$ and b) $C_{664}^*$ inside of a region of the IF lattice.
Figure 3: a) $C^{*}_{37}$’s view with nucleus n7 and b) $C^{*}_{37}$’s view with nucleus n1 IC.

Figure 4: a) $C^{*}_{37}$’s view with nucleus n1 IR and b) non optimal, symmetric $C_{37}$ with nucleus n7.

Figure 5: a) $C^{*}_{38}$’s view with nucleus n1, and b) $C^{*}_{38}$’s classical view with nucleus n6.
Figure 6: a) $C_{107}^*$ with nucleus n7, b) $C_{107}^*$’s shell

Figure 7: a) $C_{13}^*$, b) $C_{38}^*$ with their initial points inside of CB lattice
to state the growing sequence property of the optimal LJ cluster over the IC lattice, also Hoare [8] pointed out the morphology of the microclusters. It means that clusters with relative closed number of particles could have similar geometry or in other words, they belong to same lattice or they belong to the same geometrical family or they shares some similar bricks or building blocks.

Some ideas and techniques are difficult to replicate, therefore for this article, I added a simple Matlab programs to visualize my novel cluster partition and geometry, and to help for verifying my 8 categories of classification by similar number of particles considered the nucleus.

The corroboration of my results was possible because all the putative optimal LJ clusters are reported in The Cambridge Cluster Database (CCD) [17].

The next subsection depicts the notation used. Section 2 has the properties and the proposition. The subsection 2.1 depicts the technique for the creation of a partition of the cluster’s particles into layers, and subsection 2.2 depicts an heuristic for determining a cluster’s nucleus (in the appendix a Matlab program of such heuristic is depicted). Section 3 describes my version of parallel evolutionary algorithm. The next section presents the numerical results, and finally, the last section the conclusions and the future work.

1.1 Notation

Given a set \( S \), \(|S|\) is the number of elements of the set. Also if \( A[:] \) is an array, \(|A[:]|\) is the number of elements of the array. \( \emptyset \) is the empty set. \(|\cdot|\) is the norm in \( \mathbb{R}^3 \).

 Particularly, \( C_n^* \) denotes an optimal LJ cluster with \( n \) particles, and \( C_n \) denotes an arbitrary cluster with \( n \) particles.

A cluster \( C_n \) or \( C_n^* \) are sets of natural numbers, where each number correspond to a particle’s properties \( (p_i) \). For this research the particle’s properties are the particle’s 3D coordinates. \(|p_i, p_j|\) is the Euclidian distance between particles \( p_i \) and \( p_j \).

By example, \( C_n^* = \{1, 2\} \), \( p_1 = (-d^* \sqrt{2}, 0, 0) \), and \( p_2 = (d^* \sqrt{2}, 0, 0) \) where \( d^* = \sqrt{\frac{6}{2}} \) is the optimal distance for two particles under LJ potential:

\[
LJ(d) = d^{-12} - 2d^{-6}
\]

Several references explain how to build IC and FC lattices [10, 11, 16, 18]. The CB lattice is very simply is the set of points that correspond to the intersection of the parallel lines to the axes with a separation of \( d^*/2 \) from the \((0,0,0)\).

2 Properties of LJ

There are several articles about LJ potential function’s properties. The proposition 1 in [2] is repeated as proposition 2.1 in [3], together with proposition 2.2:

**Proposition 2.1** Exist a discrete set, \( \Omega \), where \( \forall j \in N, j \geq 2 \), the potential of \( \text{SOCDXX}(j) \) has the same ("close value") optimal value of \( \text{SOCCXX}(j) \) for a potential function such that

1. \( \lim_{r_{i,j} \to 0} VXX(r_{i,j}) = \infty \).
2. \( \nabla^2 VXX(x^*) \) semi-positive, \(|\nabla VXX(x^*)| < 1 \) and \( \frac{||\nabla VXX(x^*)||}{VXX(x^*)} < \delta_0 \), where \( 0 < \delta_0 \ll 1 \)

where XX is BU or LJ.

where \( \text{SOCYXX} \) means search for optimal cluster, \( \text{D} \) is discrete, \( \text{C} \) is continues and \( \text{XX} \) is LJ for Lennard Jones Potential or BU for Buckingham potential.

**Proposition 2.2** Any shape of \( n \) particles with edges \( d^* \) can be approximated from the CB lattice.

This means that with an appropriate region of CB lattice is sufficient to look for optimal clusters of size \( n \). I did not state the size of the appropriate region of CB. However, today, any optimal LJ cluster in the CCD has an initial configuration in CB lattice, such that from this initial configuration converges by a minimization process to its corresponding putative optimal LJ cluster.
Proposition 2.1. For any set of particles of a cluster’s CL or a set of particles of a region RL of any lattice based on a unit u. Then corresponding region RB of the CB lattice such RB covers them under the ||·||∞. Then particles of CL or RL are less than the number of particles of RB, i.e., |CL| < |RB| or |RL| < |RB|, where | · | is the number of particles.

Proof. Under the ||·||∞ any region of CB is a 3D cube. By construction, it has a point at the center (0,0,0), the first cube with −u/2,0,0/2 has 3³ points, the second cube with −u, −u/2,0, u/2, u has 5³ points, ..., the k cube has (2k + 1)³ points. Any polyhedra or lattice based in the unit u can not have more than 12 neighbors at ratio u. The icosahedra has 13 points but the corresponding cube to cover it is the second cube, with 5³ points, i.e., no. particles of icosahedra i no. particles of unit cube of CB lattice. In general, for a given cluster CL or region RL they can be divided and contained by a set of unit cubes of CB, which is a cube, let’s call RB. Therefore, |CL| < |RB| or |RL| < |RB|.

It follows that any region of the CB lattice has more points than the same region of a lattice.

But more important, the global continuous optimal LJ cluster can be approximated in a discrete set of 3D points, Ω. Then a connection between RB an appropriate region and Ω will be provide a discrete set of 3D points where the continuous optimal global cluster is approximated by a discrete set of points! A local minimization procedure is the connection to approximate the continuous optimal global in RB. On the other hand, for a cluster with n particles, let’s suppose to have an appropriate region RB, m = |RB|. The number of the posible clusters of size n in RB is M = \binom{n}{m} ≫ 0.

Note that M is a big number. It follows naturally from prop. 2.1 than for any region of a given lattice, the number of clusters with n particles is ≪ M.

Proposition 2.2. RB is an appropriate search region of the CB lattice for a cluster with n particles, M = \binom{n}{m}, m = |RB|, M is a huge positive number. |RL| is a region of a lattice where there are different clusters with n particles, and it is supposed that it contains the optimal LJ cluster. Then

\[ P\left(\frac{F_1}{RB}\right) < P\left(\frac{F_2}{RB \cup RL}\right) \]

where P(·) is a probability function, F₁ is the set of the optimal candidates for being the global optimal LJ cluster in RB, and F₂ is the set of the optimal candidates for being the global optimal LJ cluster in RB ∪ RL.

Proof. It follows from

\[ |RL|f_1 < f'M \]

where \( f_1 = |F_1|, f' > 0 \) is the number of candidates for being the global optimal LJ cluster in RL, \( f' > 0 \), it is not zero because the assumption that RL contains the optimal LJ cluster, \( f_1 ≪ \sqrt{M} \), and |RL| ≪ \sqrt{M}. Then

\[ (M + |RL|)f_1 < (f_1 + f')M, \]

\[ P\left(\frac{F_1}{RB}\right) = \frac{f_1}{M} < \frac{f_1 + f'}{M + |RL|} = P\left(\frac{F_2}{RB \cup RL}\right) \]

It is important to assume that \( F_2 \cap RL \neq \emptyset \), to increase the probability for determining the global optimal clusters, otherwise the probability does not increase. Many of the ad-hoc, heuristic, genetic, and evolutionary algorithms for determining the optimal LJ clusters have been used this property as previous knowledge to favor some candidates over others with success and speed to replicate the putative optimal LJ clusters.

In a personal communication, I suggested at 2004 to Shao, et al. to use different lattices from [16]. In [15] appears the acknowledge: "The authors would like to thank Prof. Carlos Barrón Romero for his personal communications and collaborations with us in the studies on the lattice-based optimization methods, including also the work published in J. Phys. Chem. A, 108, 3586-3592 (2004)."

So even, knowing that an appropriate region of the CB lattice has the optimal LJ clusters, to improve the efficiency for determining optimal LJ cluster is a good strategy to use other sources of candidates to favorece diversity in the complex process for looking the unknown optimal LJ clusters.
Finally, the theoretical results point out that it is possible to increase the speed of any algorithm for determining optimal LJ clusters but without any proof that they are global optimal. The repeated putative optimal LJ clusters are stationary states, from where a criteria such of the number of times that the same cluster appears, then stop and accept it as the putative optimal global LJ cluster.

The number of steps in these cases are clearly very less than $M = \binom{n}{m}$, $m = |RB|$. $M$ is the huge number related to the numbers of candidates to compare for determining global optimality in $RB$. A force brute algorithm for the estimation of the different combinations of a set can be found in [3], it is a version for determining the different cycles of a complete graph, $G = (V, A), |V| = n$.

2.1 Partition technique for a cluster

The geometry of the LJ clusters have been strongly related to different geometric structures (see [9, 8, 12, 16]) icosahedral, dodecahedral, cuboctahedral, and so on.

My segmentation’s technique provides different cluster’s views as an arbitrary polyhedron with its partitioning into its core, layers and shell by using the particle’s neighbors. The advantages to segment a cluster with my technique are 1) to help for interpretation and interaction with other clusters and lattices, and 2) to build a cluster from lego or building blocks.

These properties are quite important because they support the previous research about the knowledge of the clusters’ morphology, properties, geometrical families, chemistry, or the well know grow sequence.

A particle’s neighbor structure is defined as follow:

1. Define a unit: $u$.
2. Define a tolerance $t$ as the porcentaje for accepting the expansion and the compression of $u$. $(0 \leq t \leq 1.0)$.
3. Neighbor’s criteria: Particles $p_i, p_j$ are neighbor if an only if $(1-t)u < dist(p_i - p_j) < (1+ t)u$.
4. for each particle, $Nvec(i)$ is the number of neighbor of the particle $p_i$, and $Vec(i, k)$ is the array for storing the number of the particles $p_k$ that they satisfy the neighbor’s criteria with a given particle $p_i$.

where $dist$ is an appropriate distance function between the particles. $p_i$ stands for particle’s representation in a n dimensional space, a particle is represented by $p_i$, which it could contain all the relevant particle’s attributes.

For this research, $u = d^*$, $p_i$ is the particle’s 3D coordinates, $dist = || \cdot ||$ is the Euclidian distance, and $t = 0.1$. One characteristic of LJ clusters is the compression-expansion over the distance between particles with respect to the unit $d^*$. The value of $t = 0.1$ allow to differentiate a diagonal from a expanded-compressed unit $u$ and it works well to identify the ”hard LJ optimal clusters” (see [11], by example clusters with 38, 75, 98, 75, 76, 77, 102, and 103 particles). For any possible LJ cluster, the upper limit of 12 neighbors over its particles, i.e., $Nvec(i) \leq 12, \forall i.$ comes from the upper limit inherited by the 3D twelve kissing spheres geometrical property.

With the cluster’s neighbors information, the next algorithm builds an arbitrary partition of a cluster’s particles into a set of layers:

**Algorithm 2.3. Partitioning a cluster $C_n$**

**input:** $C_n$: array of int, for the set of particles’ numbers;

$Nvec$: array of int, with the particles’ number of neighbor;

$Vec[i,k]$: array of [int, int] with the particles’ neighbors;

$Nuc = \{i_0, i_1, \ldots, i_k\}$: array of int, for a given set of particles’s number to be the nucleus, with $|Nuc| \leq n$;  

**output:** $capa$: array of int, for the corresponding layer of a particle;

$ncapa$: int, for the cluster’s number of layers;

**memory:** $fmk$: int;

```plaintext
for i := 1 to n do
    capa[i] := 0;
end for
ncapa := 1;
for i := 1 to |Nuc| do
    capa[Nuc[i]] := ncapa
```
Hereafter, layer number 1 is the core or nucleus, and the last layer \( n\text{capa} \) is the shell. It is easy to verify that the set \( \text{capa} \) of the particles' numbers is a partition of \( C_n \), i.e.,:

\[
\bigcup_{i=1}^{n\text{capa}} \text{capa}[i] = C_n
\]

\[
\text{capa}[i] \cap \text{capa}[j] = \emptyset, \ i \neq j.
\]

The previous algorithm gives an arbitrary cluster partition that it could not correspond to the standard accepted geometric structures but it can be used for genetic cuts for creating a new offspring as playing with set of figures of lego. In particular, the results could be similar to the Hoare’s (see [8]) morphology of simple microclusters, polyhedral structures (PT), and an arbitrary representation of cluster’s isomeros. Figure 8 depicts \( C_{37}^* \) with its layers for an arbitrary selection of its nucleus. Figure 6 depicts \( C_{107}^* \) with nucleus \( n7 \) and its shell.

2.2 Heuristic for determining a nucleus for a cluster

The proposed heuristic is simple and it is based in previous knowledge of the LJ cluster structures that other authors have been point out.

The main concept is to look for a set of cluster’s particles inside of an sphere with center at the cluster’s center of mass with ratio 1.1\(d^*\). The particles inside of the sphere are natural candidates for being considering the cluster’s nucleus. There many cases, for the selection of the cluster’s nucleus. Let \( P\text{N} \) be the set of particles inside of the sphere, and \( cm \) the cluster’s coordinates of its center of mass:

1. IC or IR when \( \exists \ p_k \in P\text{N}, \) such that \( \arg k = \min_k ||p_i - cm||, \) and \( ||p_k - cm|| < 0.35d^* \). By example, \( C_3^3, C_5^5, \ldots \) are IC (n1 IC), and \( C_7^5, C_7^6, \ldots \) are IR (n1 IR).
2. IC without a particle as a center when \(|PN| = 12\) and these 12 are closed to the sphere’s shell. This is the case for IC nucleus with 12 particles (n0 IC). By example, \(C_5^2 21, C_5^3 33, \ldots\).

3. When \(3 \leq |PN| \leq 7\), \(PN\) is considering the cluster’s nucleus. This criteria gives a nucleus with 3 to 7 particles. By example, \(n3: 665, 668, 672, 673, 728, \ldots ; n4: 26, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 98, \ldots ; n5: 22, 23, 24, 25, 28, 29, 33, 34, 78, \ldots ; n6: 31, 32, 38, 43, 44, 49, 99, 121, \ldots ; n7: 18, 19, 20, 21, 27, 30, 35, 36, 37, 39, \ldots\).

4. when \(|PN| \geq 8\), adjust the center of the sphere to the center of mass of mass of the the cluster with the \(PN\)’s particles with 12 neighbors. The new sphere’s ratio is set to \(0.9d^*\), then take as the nucleus the particles inside of this new sphere.

It is showed in the appendix the Matlab routine "Splot_geCLJ.m". It is a version of an algorithm using this heuristic. Figure 6 depicts the results of the \(C_{107}^*\) and its shell with algorithm 2.3 with a nucleus defined by the heuristic of this section.

The next algorithm builds a set of coordinates or give a set of number that they correspond to a cluster inside of a region. A region could be and arbitrary set of points, or points of a lattice, or the points of other big cluster.

**Algorithm 2.4.** *Matching a cluster \(C_n\) with a given region \(R\)*

```plaintext
input:
p_n: array of 3D, for the particles’ coordinates of the cluster \(C_n\);
R_k: array of 3D, with the points’ coordinates of the region \(R\);
M: int, with the number of points of the region \(R\);

output:
r[i]: array of int, for the corresponding particles’ numbers in the \(R\);
s[i]: array of 3D, for the closed corresponding particles’ coordinates of the points of \(R\);

memory:
d_min: real; // minimum distance
i_min: int; // particle’ number
mk[i]: := 0: array of int, set all to 0;

if M < n then
    print("Error, it is insufficient the number of points of \(R\) for the cluster");
    return;
end if

for i := 1 to n do
    d_min := 10^8;
    for k := 1 to M do
        if (mk[k] == 0) then
            d := dist(p[i], R[k]);
            if (d < d_min) then
                d_min := d;
                i_min := k;
            end if
        end if
    end for
    mk[i_min] := 1;
    r[i] := i_min;
    s[i] := R[i_min];
end for

return;
```

The previous algorithm always answers with a set of points and with a set on particles' numbers that they correspond to a cluster, but the original and the output cluster from the region \(R\) could have very different shapes. By example, when a cluster is centered at \((0,0,0)\), and the region is box \([20, 50] \times [20, 50] \times [20, 50]\) of the cb lattice. On the other hand, figure 7 depicts \(C_{13}^*\) and \(C_{38}^*\) inside of a CB region, where the light blue small boxes are their corresponding points of CB lattice.
3 Parallel evolutionary algorithm for searching optimal LJ clusters

The novelty of the Parallel evolutionary algorithm of this section is not a new complete paradigm, as I mentioned before, it is a reload version of previous ad-hoc and genetic algorithm (see [7, 4, 5, 6, 14, 13, 2, 3]).

The term evolutionary algorithm for my approach is justified by the fact that the results are converted into input data, and this cause a change of the expected behavior of the program beyond of its programming. What, I precisely mean is that the efficacy and efficiency of the program for determining optimal LJ clusters is improved. Also, I added new routines based in phenotype and genotype strategies that my previous algorithms have not. But in the essence, it is an evolutionary program improved by his creator to increase its efficiency and efficacy with adding changes into its old routines by hand. One of the aspect to point out, is that previous version was only based in elitism, here the diversity is favored and it will come from the data of the optimal clusters and for the data of the CB lattice, particularly.

The algorithm [14] now includes the following genotype and phenotype mechanisms.

For the genotype mechanisms, a telephone model for the clusters consist to get a set of number to represent a cluster. This can be done with a region of a lattice and a cluster by using the algorithm [2.4].

An enumeration from the nucleus to its shell can be done by ordering the particles of a region by its ratio, y coordinate and its angle on the XY plane. Figure 9 depict a IC and CB regions with this numeration. This helps because for a cluster, lower number are in the core, larger number are in the shell whatever it is respect to a lattice or to other $C_n^*$.

With the telephone model for a cluster, a mutation is like to dial the cluster’s telephone with one or more mistakes. The mistake numbers can be replaced by any number not in the cluster’s telephone with the condition that such numbers are the indices of particles’ coordinates in a given region.

Children can be created by replacing sequences of the clusters’ telephones of 2 or more parent clusters.

After the creation of the children by any genetic mechanism, a minimization procedure is applied for the corresponding coordinates of the particles’ numbers of the children to get a local minimal LJ cluster for elitism selection.

On the other hand, the previous genetic algorithm adds mechanisms to use the algorithm [2.4] with or without the heuristic given in subsection [2.2]. New kind of mutations are incorporate by using the matching algorithm [2.4] to transform a cluster into a CB, IC, IF or any lattice before crossover and make up. Previously, the current population include the optimal LJ cluster, the cluster with more 12 neighbors, and the worst LJ cluster. The change is to include the lower and as many clusters as possible of the 8 categories of the heuristic for determining the nucleus with the current optimal LJ cluster.

The parallel algorithm defines a player main routine, which consists in two main routines: Cerberus and Prometheus.
Algorithm 3.1. Player

input:
  timer: set an interval of time for sending a stop signal.
  \( I_n \): int parameter of the initial cluster (\( \geq 13 \));
  \( F_n \): int parameter of final cluster (\( \geq 13 \) and (\( \geq I_n \)));
  \( P_{sz} \): int parameter of the population size (\( \geq 9 \)).
  \( R_{CB} \): set of 3D points of the CB lattice (\( \gg F_n \));
  \( R_{L1}, \ldots R_{Lk} \): set of 3D points of other lattices (\( \gg F_n \));

output:
  memory:
    \( C^*_n \): private data of the current putative optimal LJ clusters;

while (1) do
  execute \texttt{Cerberus};
  execute \texttt{Prometheus}(\( I_n, F_n, P_{sz}, R_{CB}, R_{L1}, \ldots R_{Lk} \));
  a timer or the user send a signal to stop;
end while

Cerberus is an elitism routine for communicating and keeping the best putative optimal LJ clusters. It takes care of the communication but never interrupt the process of Prometheus.

Algorithm 3.2. \texttt{Cerberus}

input:
  \( P_c \): input pile of messages for \( C_k \) (LJ clusters);
  \( \text{stp} \): int, exclusive variable to communicate with Prometheus's state, 1: Prometheus is searching or 0:Prometheus is not searching;

output:
  memory:
    \( C^*_n \): private data of the current putative optimal LJ clusters;

while (1) do
  if (\( \text{stp} == 0 \)) then
    while pile(\( P_c \)) is not empty do
      signal \texttt{Prometheus} to wait;
      \( C_k := \text{pop}(P_c) \);
      if \( LJ(C_k) < LJ(C^*_k) \) then
        \( (C_k^*) = (C_k) \);
        send message \( C^*_k \) to others players;
      end if
    end while
  end if
signal \texttt{Prometheus} to continue;
end while

Prometheus is the implementation of the previous evolutionary algorithm. It has exclusive access to the best LJ clusters during the evolutionary process.

Algorithm 3.3. \texttt{Prometheus}

input:
  \( I_n \): int parameter of the initial cluster (\( \geq 13 \));
  \( F_n \): int parameter of final cluster (\( \geq 13 \) and (\( \geq I_n \)));
  \( P_{sz} \): int parameter of the population size (\( \geq 9 \)).
  \( R_{CB} \): set of 3D points of the CB lattice (\( \gg F_n \));
  \( R_{L1}, \ldots R_{Lk} \): set of 3D points of other lattices (\( \gg F_n \));
  \text{signal}: semaphore command for waiting or executing;

output:
  memory:
    \( \text{stp} \): int, exclusive variable to communicate its state, 1: busy or 0:waiting;
Figure 10: Histogram of nucleus type for $C_n^*$, $n = 13, \ldots, 1612$ particles

Memory:

$C_n^*$: private data of the current putative optimal LJ clusters;

\begin{verbatim}
while (1) do
    stP := 0;
    for n := I_n to F_n do
        do Cerberus(signal) or player(signal);
        stP := 1;
    execute: evolutionary algorithm for exploring $C_n$
    stP := 0;
end for
end while
\end{verbatim}

When Prometheus is executing the evolutionary algorithm, there is not access to the private memory even for others clusters different of the current $n$. This is because the evolutionary algorithm could use any $C_j^*$ for creating offsprings at any time. Before or after, the process of the evolutionary algorithm, or when Prometheus is off, there is access to the best LJ optimal clusters.

The player routine is designed for working with copies of it. This could cause a bottle’s neck for the communications. Therefore, it is convenient to define a master player. In this case, only the master has the ability to send and receive messages, meanwhile the slave players can only send messages to it.

4 Results

My previous results \[2\] are in the figure 1. It depicts a set of particles MIF1739, which contains $C_n^*$, $n = 2, \ldots, 1000$. I tried to use MIF1739 has a main lattice from where an algorithm could takes advantage of its building property: $\exists C_n \in$ MIF1739, such that by a minimization process, $C_n$ converges to $C_n^*$. However, it is not easy task to locate a "good" initial set of points closed to an optimal LJ cluster. Figure 2 depicts where $C_{38}^*$ and $C_{664}^*$ are located into the IF lattice. It is possible to select points by using sphere in MIF1739. Two parameters are need, the ratio and the center of sphere.

The efficiency of the evolutionary algorithm changes dramatically with the incorporation of the CB lattice, and the phenotype and genotype strategies. The best results comes from starting with no optimal clusters but by using CB lattice and IC, IR, FC, dodecahedral lattices, and 14 as the size of the population.
Figure 11: LJ potential, potential difference vs $C^*_n$, $n = 13, \ldots, 332$ particles

Figure 12: LJ potential, potential difference vs $C^*_n$, $n = 333, \ldots, 652$ particles
Figure 13: LJ potential, potential difference vs cluster with $C_n^*$, $n = 653, \ldots, 972$ particles

Figure 14: LJ potential, potential difference vs $C_n^*$, $n = 973, \ldots, 1292$ particles
The heuristic for determining a nucleus (see subsection 2.2) and the algorithm 2.3 helped to extend the selection and interaction between clusters to create offspring for mutation or phenotype crossover or genotype crossover. This heuristic classifies into 8 categories by just using the numbers of particles in a nucleus. The nucleus types n4, n5, n6, and n7 are not geometrically equal. A refinement of heuristic is possible, but it has a computation cost.

Figure 10 depicts the histogram of 8 categories resulting of this heuristic for the $C_n^*$, $n = 13, \ldots, 1612$. Tables 1, 2, 3, and 4 contains the classification of the clusters. It seems that many cases of $C_n^*$ are obtained rapidly by a make up operation of its previous or next clusters, but also by considering to extend the diversity of the current population to 8 categories.

My results could help to answer some old conjectures about the morphology of the microclusters. See Hoare [8]: "Werfelmer’s essential contribution was to point out the possibility of extremely compact fivefold symmetric structures for N7, suggesting that the pentagonal bipyramid (N=7) (fig.3(a)) and the icosahedron (N=13) (fig.3(b)) might be the dominant motifs in larger assemblies." It is partially true, from table 4 the type 7), a nucleus with 7 particles (a pentagonal bipyramid) is the dominant motif for $C_n^*$, $n = 18, \ldots, 1530$, but IC is not a dominant motif from table 1 type 1) and type2) without considering type 8) (a nucleus IC with 12 particles, which was unknown at 1983).

Figures 11, 12, 13, 14, and 15 depict where the different nucleus type appears. At the bottom of each figure the LJ potential difference bet the consecutive clusters is depicted. Some type of optimal clusters are isolated and the LJ potential difference is highly variable from $n = 13$ to 1420. But after this cluster it seems to diminish its variations for the IR $C_n^*$, $n \geq 142$. My results are 1600 optimal LJ clusters with $n = 13, \ldots, 1612$ particles, (Most of them are posted in The Cambridge Cluster Database (CCD) [17]), a novel 8 categories of nucleus classification, and 65 new putative LJ Clusters, which are not reported at December of 2016. See table 5.

5 Conclusions and future work

The advances of the technology and science of Physics and Chemistry are fantastic, together with the molecular and the nanostructures design. The results presented here have many implications for the computational molecular design and their models and algorithms.
I hope to witness, that it is quite possible to replicate and to improve these results by using one of the top worldwide supercomputer. Some of the definitions are broad, and this research can be easily extended and applied for exploring geometries and interactions of clusters under other molecular potentials.

**Appendix**

Matlab Programs

```matlab
function S_plot_Cl_LJ(ncl,xcl,ycl,zcl,nv1,nv2)
% This subroutine draws the geometry of
% a minimal Lennard Jones Potential's cluster
% The input parameters are:
% ncl : numbers of particles
% xcl, ycl, zcl: arrays of numbers corresponding to the 3D
% cluster’s coordinates
% nv1, nv2 : integer number to determine the shells to draw
% With nv1=0 and nv2=2, the cluster’s nucleus and the first shell
% are depicted (layer 1 is the nucleus).
% ==============================================================
% More information of this subroutine is in the article:
% Discrete Optimal Global Convergence of an Evolutionary Algorithm
% for Clusters under the Potential of Lennard Jones
% Author: Carlos Barron-Romero
% Universidad Autonoma Metropolitana, campus Azcapotzalco
% Mexico City.
% This subroutine can be freely used, distributed or modified.
% ==============================================================
% Compute center of mass
xclm=mean(xcl);
yclm=mean(ycl);
zclm=mean(zcl);
% Arrays for particle’s neighbors and number of neighbors
vec=zeros(ncl,12);
nvc=zeros(ncl,1);
% dmg is the optimal distance of a pair of particles under
% Lennard Jones Potential
dmg=2^(1/6);
% 10% is the factor to define the lower and upper limits
% for accepting a particle’s neighbor
tol=0.1;
d_inf = dmg * (1 - tol);
d_sup = dmg * (1 + tol);
% This loop determines the particle’s neighbor
for i=1:ncl-1
    for j=i+1:ncl
        dij=norm([(xcl(i)-xcl(j)),(ycl(i)-ycl(j)),(zcl(i)-zcl(j))]);
        if ((d_inf < dij) && (dij < d_sup))
            nvc(i)=nvc(i)+1;
            vec(i,nvc(i))=j;
            nvc(j)=nvc(j)+1;
            vec(j,nvc(j))=i;
        end
    end
end
```

16
% Select a set of particles closed to the cluster’s center of mass
ra = dmg*1.1;
pnuc=zeros(13,1);
nnuc=0;
fnuc = 0;
for i=1:ncl
    dcl=norm([(xcl(i)-xclm),(ycl(i)-yclm),(zcl(i)-zclm)]);
    if (dcl < ra)
        nnuc=nnuc+1;
        pnuc(nnuc)=i;
    end
end
% Analyze the set of particles
if (nnuc >0)
    % First case.
    % Look for a particle with 12 neighbors
    % closed to the CM
    inuc = -1;
    d12nuc = 99999.9999;
    for k=1:nnuc
        i=pnuc(k);
        if (nvc(i) == 12)
            dcl=norm([(xcl(i)-xclm),(ycl(i)-yclm),(zcl(i)-zclm)]);
            if (dcl < d12nuc)
                d12nuc = dcl;
                inuc=i;
            end
        end
    end
    ra_nuc = dmg * 0.35;
    if ((inuc ~= -1) & (d12nuc < ra_nuc))
        % There is a particle with 12 neighbors
        pnuc(1) = inuc;
        nnuc = 1;
        ra = d12nuc;
        fnuc = 1;
    end
    % After verify that there is no center,
    % Then it is the nucleus with 12 particles
    if ((inuc == -1) & (nnuc == 12))
        fnuc = 1;
    end
end
% Final case.
% Skip four (tetrahedron, nnuc=4),
% five (Trigonal bipyramid, nnuc=5),
% six particles (octahedron, nnuc=6),
% (pentagonal polyhedron bipyramide, nnuc=7)
% if ((fnuc == 0) & (nnuc >= 8))
% Adjust the cluster’s center of mass
% considering only particles with 12 neighbors
% of the selected set pnuc
nclm = 1;
for k=1:nnuc
  i=pnuc(k);
  if (nvc(i) == 12)
    xclm = xclm + xcl(i);
    yclm = yclm + ycl(i);
    zclm = zclm + zcl(i);
    nclm = nclm + 1;
  end
end
xclm = xclm / nclm;
yclm = yclm / nclm;
zclm = zclm / nclm;
% Adjust the selected set, keeping the closed
% to the adjusted center
ra = dmg*0.9;
pnuc_nw=zeros(13,1);
nnuc_nw=0;
for k=1:nnuc
  i=pnuc(k);
  dcl=norm([(xcl(i)-xclm),(ycl(i)-yclm),(zcl(i)-zclm)]);
  if (dcl < ra)
    nnuc_nw=nnuc_nw+1;
    pnuc_nw(nnuc_nw)=i;
  end
end
% Take this new set as the nucleus
pnuc = pnuc_nw;
nnuc = nnuc_nw;
fnuc = 1;
end
% Determine the cluster's layers
% with the nucleus particles
capa=zeros(ncl,1);
capa=1;
for jp=1:nnuc
  capa(pnuc(jp))=1;
end
while (1)
  fmk=0;
  for i=1:ncl
    if (capa(i) == ncapa)
      for jv=1:nvc(i)
        pvc=vec(i,jv);
        if (capa(pvc)== 0)
          capa(pvc) = ncapa+1;
          fmk=1;
        end
      end
    end
  end
  if (fmk == 0)
    break;
  end
end
ncapa = ncapa + 1;
end

% Color table
tclr = [ 1,0,0; ...
  0,1,0; ...
  0,0,1; ...
  1,1,0; ...
  51/255, 153/255, 1; ...
  1,0,1; ...
  0.5,0.5,1; ...
  0.5,0.5,0.5; ...
  1,0.5,1; ...
  1,0.5,0.5; ...
  1,0.5,0.75; ...
  1,0.75,0; ...
  0,0,0];
hold on;
for i=1:ncl
  if (capa(i) < nv1)
    continue;
  end
  if (capa(i) > nv2)
    continue;
  end
  % cluster's lines
  for jv=1:nvc(i)
    pvc=vec(i,jv);
    if (capa(i) == capa (pvc))
      line([xcl(i),xcl(pvc)], ...
        [ycl(i),ycl(pvc)], ...
        [zcl(i),zcl(pvc)], ...
        'Color',tclr(capa(i),:), ...
        'LineWidth',1);
    end
  end
end

% particles
plot3(xcl(i), ...
  ycl(i), ...
  zcl(i), '-ko', ...
  'LineWidth',1,...
  'MarkerEdgeColor','k', ...
  'MarkerFaceColor',tclr(capa(i),:),...
  'MarkerSize',8);
end
axis equal;
grid on;
linp=sprintf('Cluster of %d particles',ncl);
title(linp);
view(45,45);
hold off;

function [n,x,y,z] = S_read_Cl_LJ(file_name)
% This subroutine reads in the format of the file from the
% The Cambridge Energy Landscape Database (\protect\vrule width0pt\protect\href{http://www-wales.ch.cam.ac.uk/}{\text{http://www-wales.ch.cam.ac.uk/}}) of a minimal cluster under Lennard Jones Potential
%
% The input parameters is the file name
% Is is ####.txt, where #### is the cluster’s number of particles
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This subroutine is in the article:
% Discrete Optimal Global Convergence of a Evolutionary Algorithm
% for Clusters under the Potential of Lennard Jones
% Author: Carlos Barron-Romero
% Universidad Autonoma Metropolitana, campus Azcapotzalco
% Mexico City.
% This subroutine can be freely used, distributed or modified under
% your own responsibility.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fid=fopen(file_name);
p = fscanf(fid,'%g %g %g', [3 inf]);
close(fid);
% x, y and z are the cluster’s coordinates
x=p(1,:);
y=p(2,:);
z=p(3,:);
% n is the cluster’s number of particles
n=length(x);

% This program calls the
% subroutine S_plot_geCl_LJ_cl to draw the geometry of
% an optimal Lennard Jones cluster
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% More information of this subroutine is in the article:
% Discrete Optimal Global Convergence of a Evolutionary Algorithm
% for Clusters under the Potential of Lennard Jones
% Author: Carlos Barron-Romero
% Universidad Autonoma Metropolitana, campus Azcapotzalco
% Mexico City.
% This subroutine can be freely used, distributed or modified under
% your own responsibility.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

feature('UseGenericOpenGL',0);
[filename,pname] = uigetfile('*.TXT');
% Define your own routine to read the particles’ coordinates
% of a optimal cluster under the Lennard Jones Potential
% or S_read_cl_LJ is set to read the files in the
% The Cambridge Energy Landscape Database (\protect\vrule width0pt\protect\href{http://www-wales.ch.cam.ac.uk/}{\text{http://www-wales.ch.cam.ac.uk/}})
%
% The input parameters is the file name
% Is is ####.txt, where #### is the cluster’s number of particles
[ncl, xcl,ycl,zcl] = S_read_Cl_LJ([pname,filename]);
nv1=0;
nv2=2;
clf;
S_plot_geCl_LJ(ncl,xcl,ycl,zcl,nv1,nv2);
References

[1] Conquering the hard cases of lennard-jones clusters with simple recipes. *Computational and Theoretical Chemistry*.

[2] C. Barrón-Romero. Minimum search space and efficient methods for structural cluster optimization. *arXiv*, Math-ph:0504030-v4, 2005. To honor the CIMAT’s XXV Anniversary.

[3] C. Barrón-Romero. The Complexity of the NP-Class. *arXiv*, arxiv.org/abs/ 1006.2218, 2010.

[4] C. Barrón-Romero, S. Gómez, and D. Romero. Archimedean Polyhedron Structure Yields a Lower Energy Atomic Cluster. *Applied Mathematics Letters*, 9(5):75–78, 1996.

[5] C. Barrón-Romero, S. Gómez, and D. Romero. Lower Energy Icosahedral Atomic Cluster with Incomplete Core. *Applied Mathematics Letters*, 10(5):25–28, 1997.

[6] C. Barrón-Romero, S. Gómez, D. Romero, and A. Saavedra. A Genetic Algorithm for Lennard-Jones Atomic clusters. *Applied Mathematics Letters*, 12:85–90, 1999.

| n1 IC | n1 IR |
|-------|-------|
| 13 14 15 16 17 45 46 47 48 49 | 75 76 77 188 189 190 191 192 650 651 652 653 654 655 |
| 50 51 52 53 54 55 56 57 58 59 | 656 657 658 659 660 661 662 666 682 683 684 685 |
| 60 61 62 63 64 65 66 67 126 | 686 687 688 689 691 1027 1029 1031 1033 1035 1036 |
| 127 128 129 130 131 132 133 | 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 |
| 134 135 136 137 138 139 140 | 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 |
| 141 142 143 144 145 146 147 | 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 |
| 148 149 150 151 152 153 154 | 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 |
| 155 156 157 158 159 160 161 | 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 |
| 162 163 164 165 166 168 272 | 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 |
| 273 274 275 276 277 278 279 | 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 |
| 280 281 282 283 284 285 286 | 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 |
| 287 288 289 290 291 292 293 | 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 |
| 294 295 296 297 298 299 300 | 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 |
| 301 302 303 304 305 306 307 | 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 |
| 308 309 310 311 312 313 314 | 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 |
| 315 316 317 318 319 320 321 | 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 |
| 322 323 324 325 326 327 328 | 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 |
| 329 330 331 332 333 334 335 | 1177 1178 1179 1180 1181 1182 1183 1184 1185 1186 |
| 336 337 338 339 340 495 498 | 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 |
| 495 503 504 505 507 508 509 | 1197 1198 1199 1200 1201 1202 1203 1204 1205 1212 |
| 510 511 512 513 514 515 516 | 1213 1214 1215 1216 1217 1218 1219 1220 1219 1515 1516 |
| 517 518 519 520 522 523 524 | 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 |
| 525 526 527 528 529 530 531 | 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536 1537 |
| 532 533 534 535 539 540 544 545 | 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 |
| 549 550 551 552 553 554 555 | 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 |
| 556 557 558 559 560 561 562 | 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 |
| 563 564 565 566 567 568 569 | 1568 1569 1570 1571 1572 1573 1574 1575 1576 1577 |
| 570 571 572 573 574 575 576 | 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 |
| 577 578 579 580 581 582 583 | 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 |
| 584 586 587 588 589 590 591 | 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 |
| 592 593 594 595 596 598 599 | 1608 1609 1610 1611 1612 |
| 600 601 602 603 604 923 | |

Table 1: Type 1 and 2 of the $C_n^a$, $n = 13, \ldots, 1612$
| n3  | 29 |
|-----|----|
| 665 | 668 |
| 728 | 729 |
| 733 | 734 |
| 738 | 739 |
| 743 | 744 |
| 748 | 749 |
| 26  | 86  |
| 207 | 208 |
| 395 | 396 |
| 410 | 411 |
| 710 | 711 |
| 852 | 853 |
| 29  | 91  |
| 668 | 672 |
| 731 | 732 |
| 736 | 737 |
| 741 | 742 |
| 747 | 751 |
| 854 | 855 |
| 673 | 674 |
| 678 | 680 |
| 712 | 713 |
| 755 | 756 |
| 760 | 761 |
| 847 | 848 |
| 842 | 843 |
| 858 | 859 |
| 90  | 91  |
| 210 | 211 |
| 399 | 400 |
| 414 | 415 |
| 715 | 716 |
| 885 | 886 |
| 958 | 959 |
| 90  | 91  |
| 215 | 216 |
| 401 | 402 |
| 416 | 417 |
| 717 | 718 |
| 886 | 887 |
| 963 | 964 |
| 91  | 92  |
| 217 | 218 |
| 403 | 404 |
| 418 | 419 |
| 718 | 719 |
| 887 | 888 |
| 965 | 966 |
| 92  | 93  |
| 218 | 219 |
| 405 | 406 |
| 419 | 420 |
| 719 | 720 |
| 888 | 889 |
| 966 | 967 |
| 93  | 94  |
| 219 | 220 |
| 406 | 407 |
| 420 | 421 |
| 720 | 721 |
| 889 | 890 |
| 967 | 968 |
| 94  | 95  |
| 220 | 221 |
| 407 | 408 |
| 421 | 422 |
| 721 | 722 |
| 890 | 891 |
| 968 | 969 |
| 95  | 96  |
| 221 | 222 |
| 408 | 409 |
| 422 | 423 |
| 722 | 723 |
| 891 | 892 |
| 969 | 970 |
| 96  | 97  |
| 222 | 223 |
| 409 | 410 |
| 423 | 424 |
| 723 | 724 |
| 892 | 893 |
| 970 | 971 |
| 97  | 98  |
| 223 | 224 |
| 410 | 411 |
| 424 | 425 |
| 724 | 725 |
| 893 | 894 |
| 971 | 972 |

Table 2: Type 3 and 4 of the $C_n^*, n = 13, \ldots, 1612$

| n5  | 163 |
|-----|-----|
| 22  | 23  |
| 24  | 25  |
| 28  | 29  |
| 33  | 34  |
| 78  | 79  |
| 80  | 81  |
| 267 | 268 |
| 269 | 271 |
| 486 | 487 |
| 488 | 489 |
| 490 | 491 |
| 492 | 493 |
| 494 | 496 |
| 497 | 597 |
| 606 | 612 |
| 614 | 616 |
| 619 | 620 |
| 623 | 624 |
| 634 | 636 |
| 690 | 692 |
| 694 | 695 |
| 696 | 697 |
| 698 | 699 |
| 701 | 702 |
| 703 | 704 |
| 705 | 706 |
| 707 | 708 |
| 714 | 716 |
| 718 | 719 |
| 721 | 722 |
| 723 | 724 |
| 725 | 726 |
| 777 | 778 |
| 779 | 780 |
| 781 | 782 |
| 783 | 784 |
| 785 | 786 |
| 787 | 788 |
| 789 | 790 |
| 791 | 792 |
| 793 | 794 |
| 795 | 796 |
| 797 | 798 |
| 799 | 800 |
| 801 | 802 |
| 803 | 804 |
| 805 | 806 |
| 807 | 808 |
| 809 | 810 |
| 811 | 812 |
| 813 | 814 |
| 817 | 826 |
| 827 | 828 |
| 973 | 974 |
| 975 | 976 |
| 977 | 978 |
| 979 | 980 |
| 981 | 982 |
| 983 | 984 |
| 985 | 986 |
| 987 | 988 |
| 989 | 992 |
| 993 | 994 |
| 995 | 996 |
| 997 | 998 |
| 999 | 1000 |
| 1001 | 1002 |
| 1003 | 1004 |
| 1005 | 1006 |
| 1007 | 1008 |
| 1009 | 1010 |
| 1011 | 1012 |
| 1013 | 1014 |
| 1015 | 1016 |
| 1017 | 1018 |
| 1019 | 1020 |
| 1021 | 1022 |
| 1023 | 1025 |
| 1028 | 1030 |
| 1032 | 1034 |
| 1315 | 1316 |
| 1317 | 1318 |
| 1319 | 1320 |
| 1321 | 1322 |
| 1323 | 1324 |
| 1325 | 1326 |
| 1327 | 1328 |
| 1329 | 1330 |
| 1331 | 1332 |
| 1333 | 1335 |
| 1336 | 1347 |

Table 3: Type 5 and 6 of the $C_n^*, n = 13, \ldots, 1612$
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |  

Table 4: Type 7 and 8 of the $C_\ast_n$, $n = 13, \ldots, 1612$
\[
(n, \text{ LJ}_{\text{old}}, \text{ LJ}_{\text{new}}) \quad 65
\]

| \text{LJ}_{\text{old}} | \text{LJ}_{\text{new}} | \text{LJ}_{\text{new}} | \text{LJ}_{\text{new}} | \text{LJ}_{\text{new}} | \text{LJ}_{\text{new}} |
|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| (293,-1888.4271,-1888.4274) | (506,-3427.6212,-3427.6875) | (521,-3539.3314,-3539.5098) | (533,-3628.2529,-3629.2999) | (662,-4581.2049,-4581.2058) | (664,-4596.1971,-4596.1978) |
| (1102,-3427.6212,-3427.6875) | (521,-3539.3314,-3539.5098) | (533,-3628.2529,-3629.2999) | (662,-4581.2049,-4581.2058) | (664,-4596.1971,-4596.1978) | (1102,-3427.6212,-3427.6875) |
| (1102,-3427.6212,-3427.6875) | (521,-3539.3314,-3539.5098) | (533,-3628.2529,-3629.2999) | (662,-4581.2049,-4581.2058) | (664,-4596.1971,-4596.1978) | (1102,-3427.6212,-3427.6875) |

Table 5: New \( C^* \)

[7] S. Gómez and C. Barrón-Romero. The Exponential Tunneling Method. Technical Report Research Report 3(1), IIMAS-UNAM, Julio 1991 1991.

[8] M. R. Hoare and J. A. McInnes. Morphology and statistical statics of simple microclusters. *Advances in Physics*, 32(5):791–821, 1983.

[9] M. R. Hoare and P. Pal. Physical cluster mechanics: statistical thermodynamics and nucleation theory for monatomic systems. *Advances in Physics*, 24(5):645–678, 1975.

[10] R. H. Leary. Global Optima of Lennard-Jones Clusters. *Journal of Global Optimization*, 11(1):35–53, 1997.

[11] R. Maier, J. Rosen, and G. Xue. A discrete-continuous algorithm for molecular energy minimization. In *Proceedings. Supercomputing ’92. (Cat. No.92CH3216-9)*, 16-20 Nov. 1992. Proceedings. Supercomputing ’92. (Cat. No.92CH3216-9), pages 778–786, Minneapolis, MN, USA, 1992. IEEE Comput. Soc. Press.

[12] J. A. Northby. Structure and binding of Lennard-Jones clusters: 13 ≤ n ≤ 147. *Journal of Chemical Physics*, 87(10):6166–6177, 1987.

[13] D. Romero, C. Barrón-Romero, and S. Gómez. The optimal configurations of LJ atomic clusters: 148-309. In *Siam Annual Meeting*, Atlanta, GA, USA, 1999.

[14] D. Romero, C. Barrón-Romero, and S. Gómez. The optimal geometry of Lennard-Jones clusters: 148-309. *Computer Physics Communications*, 123:87–96, 1999.

[15] X. Shao, Y. Xiang, and W. Cai. Structural Transition from Icosahedra to Decahedra of Large Lennard-Jones Clusters. *Personal Communication*, 2005.

[16] I. A. Solov’yov, A. V. Solov’yov, and W. Greiner. Fusion process of Lennard-Jones clusters: global minima and magic numbers formation. *ArXiv Physics e-prints*, pages 1–47, 2003.
[17] D. J. Wales, J. P. K. Doye, A. Dullweber, M. P. Hodges, F. Y. Naumkin, F. Calvo, J. Hernández-Rojas, and T. F. Middleton. The cambridge cluster database, lennard-jones clusters, http://www-doye.ch.cam.ac.uk/jon/structures/lj.html.

[18] Y. Xiang, H. Jiang, W. Cai, and X. Shao. An Efficient Method Based on Lattice Construction and the Genetic Algorithm for Optimization of Large Lennard-Jones Clusters. *Journal of Physical Chemistry A*, 108(16):3586–92, 2004.
