J. Indian Chem. Soc.,
Vol. 90, July 2013, pp. 879-889

ENDOWMENT LECTURE

Exploring new computing paradigms in theoretical chemistry†

Kanchan Sarkar[a] and S. P. Bhattacharyya[b]*

[a]Department of Physical Chemistry, Indian Association for the Cultivation of Science, Jadavpur, Kolkata-700 032, India

[b]Department of Chemistry, Indian Institute of Technology, Powai, Mumbai-400 076, India

E-mail : pcspb@chem.iitb.ac.in

Manuscript received 25 March 2013, accepted 26 March 2013

Abstract : Computing in theoretical chemistry has been largely and traditionally based on purely numerical 'non-intelligent' computing techniques. The tools of 'Artificial Intelligence' (AI) or 'Computational Intelligence' have been little explored and exploited in the context of research in theoretical chemistry. Over the last decade and a half we had been experimenting with 'evolutionary computing techniques' like the Genetic Algorithms and Random Mutation Hill Climbing in the general context of computing electronic structure of atoms and molecules. These methods have the underpinning of certain microscopic low-level biological processes and are supposed to be endowed with 'Artificial Intelligence'. We trace the evolution of the AI-based techniques developed by us and review some of the rather non-trivial applications. In particular, we focus on an Adaptive Random Mutation Hill Climbing (ARMHC) method for locating global minima on the complex potential energy landscapes of 3-D Coulomb clusters and assessing the possibilities of structural phase transitions in them. Possible directions of future developments are indicated.

Keywords : Computational intelligence, soft computing, evolutionary computing, adaptive random mutation hill climbing, genetic algorithms, electronic structure calculation, constrained variational calculation.

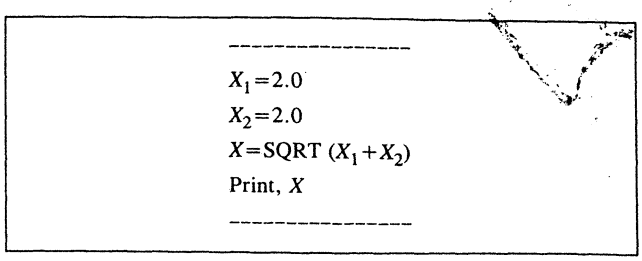
Introduction

Computers have now pervaded every sphere of our daily life, from booking of air tickets to predicting trends in share markets. It is more so when we consider the arena of scientific research. Irrespective of the specific field, the computer is ubiquitous. It could be assisting us in doing literature search, in discovering trends or systematic relationship in a huge body of unorganized data, in solving large systems of algebraic or differential equations, recognizing and classifying patterns, and so on. Bulk of it, however, belongs to what is technically called 'non-intelligent computing'. It may involve doing billions of additions, subtractions, multiplications and divisions at a very high speed – but there is nothing intelligent about it. 'Computational intelligence' or 'Artificial intelligence' is something very different from what 'traditional numerical computing' is about. Let us spend some time in tracing the evolution of what has now come to be known as 'Artificial intelligence' or 'Computational intelligence' of one kind or the other.

It was 1760 when England was swept over by the Industrial Revolution in the aftermath of which human muscle power was largely replaced by machine power in huge production systems. Almost two hundred years later (1950) the idea of replacing human intelligence by machine intelligence was first mooted. The idea aroused interest and initial excitement, but the scale of its impact was never comparable to that of industrial revolution. The term 'Artificial intelligence' first appeared in 1956 and stayed on, although the precise meaning of the term has undergone slow transmutation over the years. Going by the lexicographers' definition, the term 'Artificial intelligence' refers to 'the ability of computers to mimic human thought process such as reasoning and learning'. In a broader sense, AI may be defined as 'the ability of computers or computer programs to execute tasks which

†**Professor J. N. Mukherjee Memorial Lecture (2011) delivered in 'Acharya Prafulla Chandra Ray Memorial Symposium on Chemistry & Industry (2012)' organized by the Indian Chemical Society held at Kolkata on August 02, 2012.**

humans require intelligence to accomplish. The broader definition naturally includes the more restricted definition of AI given by lexicographers. Let us consider a simple program



The computer prints the value of X correctly as 2.0. However, it is not to be counted as an example of machine intelligence or artificial intelligence. On the other hand, if the computer can correctly find out what $\int f(x)dx$ is and print out the analytical form of the anti-derivative $F(X)$, we will identify it as a signature of artificial intelligence. Indeed, the initial emphasis on research in AI was largely on symbolic machine learning. In the traditional sense broad classes of problems that require 'machine intelligence' for finding solutions include knowledge based reasoning and making inferences, reasoning with incomplete (perhaps uncertain) information, perception and learning in various forms and their applications to problems of control, classification, prediction, optimization, etc. A second characterization of 'intelligent computing' refers to the use of microscopic (low level) biological reasoning or mechanism or models to arrive at a solution. Artificial Neural Networks and Genetic algorithms or evolutionary computing belong to the non-traditional category of intelligent computing. The focus of this lecture is on the second category of intelligent computing with emphasis on genetic algorithms in the context of solving a number of problems of theoretical chemistry.

Biological life appeared on the earth probably as a culmination of many chance events involving chemical and physical interactions of molecules. Ever since the appearance of unicellular life, progressively complex life-forms have evolved. The process of genetic evolution has elements of adaptive learning built into it. Genetic algorithms[1] are mathematical models of genetic evolution implemented on a computer and are the most versatile problem-solvers. In a broader sense, GAs are a subject of what has been called evolutionary computing techniques which include genetic programming, classifier systems, evolvable hardware, evolutionary robotics, ant colony optimization and swarm intelligence.

In a diploid organism like the human beings a pair of parental chromosomes form the chromosome-pair of the child, each chromosome containing millions of genes. One gene from the father and the corresponding gene from the mother constitute a gene-pair for the child, the gene-pair (genotype) determines the specific attributes or characteristics called the phenotypes. Most gene values (alleles) are inherited by the child from the parents unaltered; but on rare occasions, one or more of them may undergo changes, by a process called mutation. If such changes are beneficial for the individual for better adaptation to the environment, they have higher probabilities of survival, and have correspondingly higher probabilities for producing their offspring. Over generations those beneficial changes tend to stay on while the non-beneficial ones tend to disappear. There is thus a natural process of screening in the course of genetic evolution, which finds expression in the Darwinian dictat of the survival of the fittest. The evolution can be visualized as a continuous process by which a species continuously strives to attain a genetic structure of the chromosome that maximizes its probability of survival in a given environment.

Genetic algorithms

Holland[1] first proposed a mathematical model of the genetics of evolution, which could be implemented on a computer to solve hard problems. The resulting technique, called the GA, can be thought of as involving the following steps.

(1) First reduce the problem at hand to an equivalent maximization problem. The reduction leads to the definition of fitness function F that depends on the variables of the problem.

(2) Give the potential solution a chromosomal representation. The solutions are represented as binary strings (with 0, 1 as alphabets), integer strings (integer arrays), floating point strings (floating point arrays), matrices, patterns, graphs, etc.

(3) Generate a population (n_p) of such solutions $\{S_k\}_{k=1,np}$ based on random guesses or educated guesses.

(4) Evaluate the fitness $\{f_k\}$ of the n_p individuals

$f_k = F(S_k),\ k = 1, n_p$

$$f_{av} = \frac{1}{n_p}\sum_{k=1}^{n_p} f_k$$

(5) Screen the population by a fitness-proportional selection mechanism. The frequency of selection of individuals with higher fitness values is higher, so the f_{av} increases.

(6) Breed new solutions by a crossover process with a crossover probability p_c. The process can be schematically represented as follows :

Randomly choose a pair of chromosomes say, S_1 and S_2 from the post-selection population

$S_1 = (a, b, \overset{\Downarrow}{c}, d, e, f)$

$S_2 = (a', b', \overset{\Downarrow}{c'}, d', e', f')$

and choose a crossover site marked by (⇓) randomly with a probability p_c. Let the site chosen be the 3rd from the left. Then the crossover generates a pair of new strings, S'_1 and S'_2 where

$S'_1 = (a, b, c, d', e', f')$

$S'_2 = (a', b', c', d, e, f)$

It is easy to see that the strings produced by crossover will have, more often than not, new genetic structure (distribution of genes) and therefore have fitness values (f'_1, f'_2) different from the parent strings with fitness values (f_1, f_2). Once n_p number of offsprings has been produced (constant population size model), the post-crossover strings are allowed to undergo a process called mutation.

(7) A mutation probability (p_m) is fixed and one loops through each gene in a chromosome (S'_k) and mutates the gene at any site for which a random number in the interval (0,1), $r(0,1) < p_m$.

(8) In case one is pursuing a completely non-elitist strategy, the population of offsprings replaces the entire parent population. In the other extreme one can choose to be perfectly elitist and create a new population of the best n_p-strings from the parents and offsprings. One can have a continuous variation between the two extremes. Once the choice is made and the process of screening is over, it marks the passage of one generation.

The whole procedure (1–8) is repeated till the fitness of the best string in the population does not change any further or a preset number of generations have elapsed. Usually the fitness evolution profile for the best string in the population will grow exponentially (Fig. 1) with saturation as the number of elapsed generations $n_g \to \infty$ (in the elitist strategy) and will grow exponentially with oscillations (in the non-elitist strategy) and saturate in the limit $n_g \to \infty$. The superindividual is then accepted as the best and robust solution of the problem.

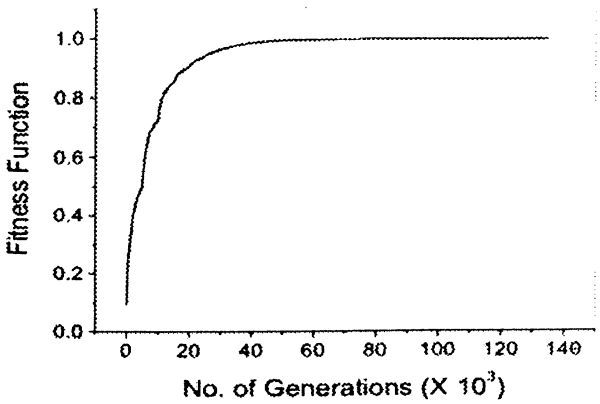


Fig. 1. Fitness evolution profile for the best string in the elitist strategy.

It is important now to reflect on the question "why do the GAs behave the way they do?" Let us consider a population of n-bit strings (n = 300, say) that are randomly chosen as plausible solutions of the given problem. For 300-bit strings, there are 2^{300} possibilities out of which one may represent the actual solution. The standard population size varies in the range 20–50, so the actual number of strings evaluated is just 20–50 at each generation. The question that immediately crops us is how does the algorithm home on to the actual solution string in a few hundred or a few thousand generations? The algorithm must be discarding the non-viable region of the search space quickly as it explores the space, based on the past experience of exploration, although nothing like that has been explicitly grafted into the algorithm. The widely accepted theoretical model that explains the intelligent behavior of the GA has been known as the Schema theorem[1]. According to the Schema theorem, GA explores the search space by implicitly and parallely keeping track of an explosively large number of templates or partial structure (Schemata) present in the population of

n_p-strings. The algorithm does not explicitly evaluate any of the partial structures. It is the parallelism between the actual evaluation of n_p-strings for their fitness and implicit assessment of the fitness of a much larger number of Schemata that helps the GA quickly move to the viable region of the search space[2]. Let us consider an example.

Supposing that we are working with 6-bit strings (2^6 possibilities) and a population of 3-strings S_1, S_2 and S_3 where

S_1 = 1 0 0 1 0 1

S_2 = 0 0 1 1 0 1

S_3 = 0 1 0 1 0 1

Let the true solution be represented by S_0 = (111111). For such a system H_1 = (* * 1 1 0 *) is one possible schema where '*' denotes undefined bits. Clearly, this partial structure is present in the string S_2 which is therefore an instance of the schema H_1. The average fitness f_{av} of the population is $\frac{1}{3}(f_1 + f_2 + f_3)$ while the fitness f_{H_1} of the schema H_1, for example, is the average fitness of all the instances of the schema in the population. We emphasize that GA evaluates f_1, f_2, f_3 explicitly in every generation but does not evaluate f_{H_1}, f_{H_2},... etc. at no stage explicitly. Two other important features of schema are its order (O_H) and distance or defining length δ_H. The order of the schema simply represents the number of defined bits (4, for example in H_1) while δ_H is the distance between first and last defined bits (δ_{H_1} = 2, for example). With these quantities defined, one can easily show that the number of instances (n_H) of the schema H of defining length δ_H, order O_H and fitness f_H in $(t+1)$-th generation ($n_H(t+1)$) grows as

$$n_H(t+1) \geq \frac{f_H}{f_{av}}\left(1 - \frac{\delta_1}{l-1} p_c\right)(1 - p_m)^{O_H} n_H(t) \quad (1)$$

where p_c and p_m are fixed crossover and mutation probabilities, respectively, $n_H(t)$ being the number of instances of the schema H in the preceding (t-th) generation[1,2]. The equation leads to the building block hypothesis which states that the GA explores the search space by emphasizing on short, low order schemata of above average fitness through the process of crossover, mutation and selection. Crossover brings better schemata (of higher f_H) on the same string while mutation helps in producing new beneficial schemata and restore good genetic material that could have been otherwise lost in the selection process. Schemata with below average fitness die off. Many computer experiments have vindicated the validity of BBH[3].

With this background about how and why a simple GA works, we may now consider a specific problem and specific GA (floating point rather than binary) for handling the problem.

Model problem :

Let us assume that our problem is to solve the linear system

$$AX = b \quad (2)$$

with $A(nxn)$ and $b(nxl)$ fixed. The solution X(nxl), if A is invertible can be straightforwardly written as

$$X = A^{-1}b \quad (3)$$

However, for large n, the direct inversion of A is computationally costly with computation labor growing as $\sim n^3$. It is possible to cast the problem in the mould of an equivalent maximization problem, which the GA can handle with great efficiency. Let us consider a population (n_p) of n-dimensional solution vector $X^{(1)}$, $X^{(2)}$, ..., $X^{(np)}$ guessed randomly. $(Ax^{(i)}-b) = \varepsilon_i$ is then the i-th error vector (nxl). The length of the error vector is $l_i=(\varepsilon_i{}^+ \varepsilon_i)^{0.5}$. For the true solution vector $l_i = 0$, while for all others $l_i > 0$. One can assign a fitness value f_i to each $X^{(i)}$ through l_i as follows :

$$f_i = e^{-l_i c} \quad (4)$$

where λ is a positive scalar chosen by the user. f_i will be in the range (0,1), with $f_i = 1$ only when $Ax_i = b$ i.e. when the length of the error vector $\varepsilon_i = 0$. We may now invoke GA to look for a vector $X^{(i)}$ in the evolving population for which the fitness f_i is maximized, which is equivalent to saying that $\varepsilon_i = 0$ or $Ax^{(i)} = b$. It is convenient to work with vectors $X^{(i)}$, $X^{(j)}$, $X^{(k)}$, etc. which have components represented by real or floating point numbers. However, we must define suitable crossover and mutation operations in the floating-point representation.

Let us suppose that the pair of strings randomly chosen for crossover operation are $X^{(j)}$ and $X^{(k)}$

where

$$X^{(j)} = (x_1{}^{(j)}, x_2{}^{(j)}, x_3{}^{(j)}, \ldots, x_i{}^{(j)}, x_n{}^{(j)})$$

and (5)

$$X^{(k)} = (x_1{}^{(k)}, x_2{}^{(k)}, x_3{}^{(k)}, \ldots, x_i{}^{(k)}, x_n{}^{(k)})$$

and suppose that the i-th site has been chosen for crossover with a probability P_c. Then the post-crossover string will be

$$X^{(j)'} = (x_1^{(j)}, x_2^{(j)}, x_3^{(j)}, \ldots, x_i^{(j)}, x_{i+1}^{(j)'}, \ldots, x_n^{(j)'}) \quad (6)$$
$$X^{(k)'} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \ldots, x_i^{(k)}, x_{i+1}^{(k)'}, \ldots, x_n^{(k)'})$$

where $x_p^{(j)'} = \sin^2\theta x_p^{(j)} + \cos^2\theta x_p^{(k)}$

and

$$X_p^{(k)'} = \cos^2\theta x_p^{(j)} + \sin^2\theta x_p^{(k)} \quad (7)$$

with θ chosen randomly in the interval $(0\text{-}2\pi)$ and $p = i + 1, i + 2,\ldots,n$. The scheme of crossover has been called arithmetic single point crossover. An arithmetic mutation operation can be similarly defined. For the post-crossover string $X^{(1)}$ where

$$X^{(l)} = (x_1^{(l)}, x_2^{(l)}, x_3^{(l)}, \ldots, x_i^{(l)},\ldots, x_n^{(l)} \quad (8)$$

If the site $x_i^{(l)}$ is chosen to undergo mutation with a probability p_m and intensity Δ_m, we have

$$x^{(l)'} = x_i^{(l)} + (-1)^L \Delta_m r \quad (9)$$

where L is a randomly chosen integer, Δ_m is a small user-defined increment in x_i^1 and r is a random number between $(0, 1)$. We have made use of Roulette-wheel or tournament selection and elitist screening. We note here, that there could be many variations on screening, selection, crossover and mutation and the optimal choice may well be problem dependent.

Applications

The basic GA-based strategy outlined in this section has been with necessary modifications used to handle the following non-trivial problems of electronic structure calculation.

(a) Direct solution of Schrodinger equation (SE) for one and two electron atoms and ions[4–7].

(b) Direct solution of molecular SE for H_2^+ ion[8].

(c) Evolution of molecular electronic structure of doped and undoped polythiophene and selenophene oligomers using a modified Su-Schrieffer-Heeger Hamiltonian based description[9,10].

(d) Constrained variational calculation of 'zero band-gap' constrained thiophene oligomers[11].

(e) Elucidation of 3D structures of coulomb crystals, etc.[12].

In what follows we present an overview of applications in the categories (a)-(c) and the progress made in handling problems in the categories (d)-(e).

(a) Direct solution of SE : Let $\psi_0(r_1, r_2)$ represent the radial wave function for the ground state of a two electron atom or ion. $\psi_0(r_1, r_2)$ then satisfies the following energy eigenvalue equation

$$H\psi_0(r_1, r_2) = E_0\psi_0(r_1, r_2) \quad (10)$$

where,

$$H = -\frac{h^2}{2m}\sum_{i=1}^{2}\left(\frac{\partial^2}{\partial r_i^2} + \frac{2}{r_i}\frac{\partial}{\partial r_i}\right) - \quad (11)$$

The total wavefunction for the ground state, a spin singlet, is

$$\Psi_0(r_1, r_2, s_1, s_2) =$$
$$\Psi_0(r_1, r_2) \times \eta_{spin}(s_1, s_2, s = 0) \quad (12)$$

The $\eta_{spin}(s_1, s_2, s = 0)$ is antisymmetric with respect to interchange of the two spin-coordinates which automatically demands $\Psi_0(r_1, r_2)$ to be symmetric under the interchange of the radial coordinates (r_1, r_2) so that Ψ_0 remains antisymmetric under the interchange of space-spin coordinates of the two electrons. We have used only $l=0$ type of one-electron wave functions in the basis which results in the following structure of the wave function strings $(k = 1, 2,\ldots\ldots n)$

$$\Psi_{(k)}(r_1, r_2) =$$
$$\sum_{ij} c_{ij}\phi_{ij}^{(k)}\left(r_1, r_2, n_i, n_j, \alpha_i^{(k)}, c\right. \quad (13)$$

where,

$$\phi_{ij}^{(k)} = N_{ij}[\chi(r_1, \alpha_{n_i}^{(k)})\chi(r_2, \alpha_{n_j}^{(k)}) + \chi(r_1, \alpha_{n_j}^{(k)})\chi(r_2, \alpha_{n_i}^{(k)})] \quad (14)$$

N_{ij} being the normalization constant for $\phi_{ij}^{(k)}$.

The wave function strings for the problem contain the linear (C_{ij}) and nonlinear parameters $\{\alpha_i, \alpha_j\}$ as information and the population of wave function strings has, for example, members like

$$\Psi_k = S_k(c_1^{(k)}, c_2^{(k)}, \ldots, c_n^{(k)}, \alpha_1^{(k)}, \alpha_2^{(k)}, \alpha_m^{(k)})$$
$$\vdots \quad (15)$$
$$\Psi_l = S_l(c_1^{(l)}, c_2^{(l)}, \ldots, c_n^{(l)}, \alpha_1^{(l)}, \alpha_2^{(l)}, \alpha_m^{(l)})$$

In a preliminary application with a population of 10 strings each encoding 21 linear and 5 nonlinear parameters Saha *et al.*[8] showed that the GA-based strategy could successfully predict H^- ion to be bound by about 0.012

a.u. The growth of the fitness value of the best string in the population as function of generations elapsed is shown in Fig. 2a, while the optimization of energy along the path of evolution is displayed in Fig. 2b.

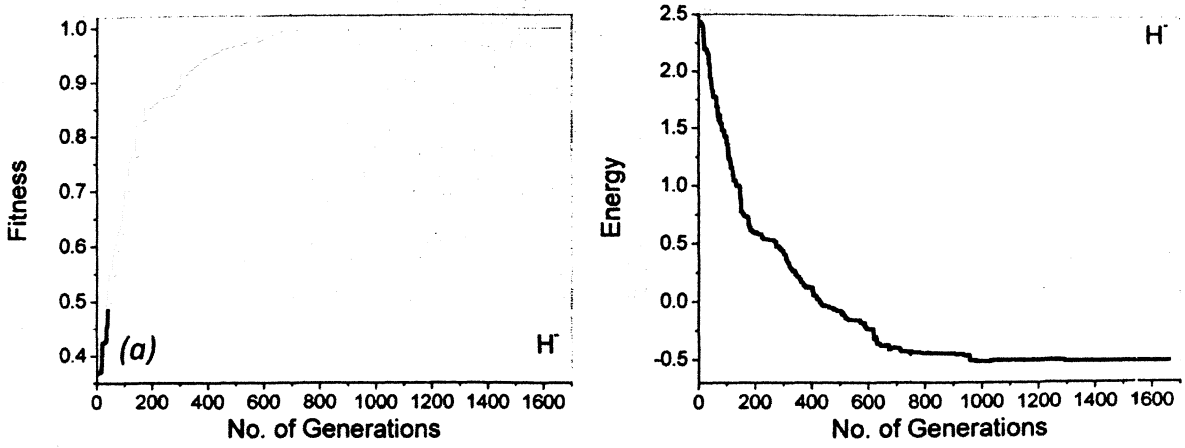


Fig. 2. Evolution profile of (a) fitness function, (b) energy for the ground-state energy of the H^- ion during a GA run.

The applications referred to in the preceding paragraph make use of wave-function strings containing all the parameters of the basis functions (Φ_{ij}) in terms of which $\Psi(r_1, r_2)$ is expanded. We can do better and do away with the basis set altogether and consider wave-function strings encoding the probability amplitude distributions in the discretized radial space (r_1, r_2) of the two-electron wavefunctions. If a grid of $n_1 \times n_2$ points is used to represent the amplitude distribution in space, we have

$$\Psi_k (r_1, r_2) \equiv S_k (n_1 \times n_2) = S_k (500 \times 500) \quad (16)$$

where the array S_k encodes $n_1 \times n_2$ values $\{S_k(i, j)\}$ such that

$$\{S_k(i,j)\}^2 = |\Psi_k (r^i_1, r^j_2)|^2 \quad (17)$$

and

$$S_k(j, i) = S_k(i, j) \quad (18)$$

The condition implied by eq. (18) ensures that the probability amplitude distribution conforms to the condition that the space-part of the wave function be symmetric under $1 \leftrightarrow 2$ interchange. For performing quadratures, additional grid-point amplitudes were generated by bi-cubic interpolation. The fitness function used was $f_k = e^{-\sigma F_k}$, where

$$F_k = \left(\frac{< \Psi_k | H | \Psi_k >}{< \Psi_k | \Psi_k >} - E_L \right)^2 \quad (19)$$

EL being an estimated lower bound to the ground state energy of the two-electron atom under consideration. For a number of two electron atoms and ions, the method predicted ground state energies lower than the corresponding Hartree Fock energy[7].

(b) Ground state of H_2^+ : The strategy was later extended to the calculation of the electronic structure of H_2^+ ion under the Born-Oppenheimer approximation[8]. The molecular electronic wave function of H_2^+ at a given internuclear separation i.e. $\Psi(x,y,z,R)$ was represented as a string of probability amplitude distribution $S(n_1 \times n_2 \times n_3)$ in 3D space such that

$$S(i, j, k)_R = \Psi_R(x_i, y_i, z_i) \quad (20)$$

n_1, n_2, n_3 were each chosen to be 128. A population of 10 strings was used with arithmetic crossover and mutation operations. Selection was through the Roulette-Wheel method. Two different schemes were worked out.

(i) In the first approach, R was kept fixed and the electronic probability amplitudes $S(i, j, k)_R$ were allowed to evolve genetically. The kinetic energy $<T>$ was evaluated by adopting discrete fast Fourier transform while the potential energy $<V>$ was obtained through quadrature by trapezoidal rule. When the evolution converged, the energy $E(R)$ corresponding to the best string was converted into the total energy $U(R) = E(R) + e^2/R$. Calculations at different 'R' then led to the full potential energy curve (Fig. 3a), from which the equilibrium R ($=R_e$) and $V(R_e) = V_e$ were computed (Type I).

(ii) In the second approach, R was allowed to evolve simultaneously with the electronic probability amplitudes so that when the genetic evolution

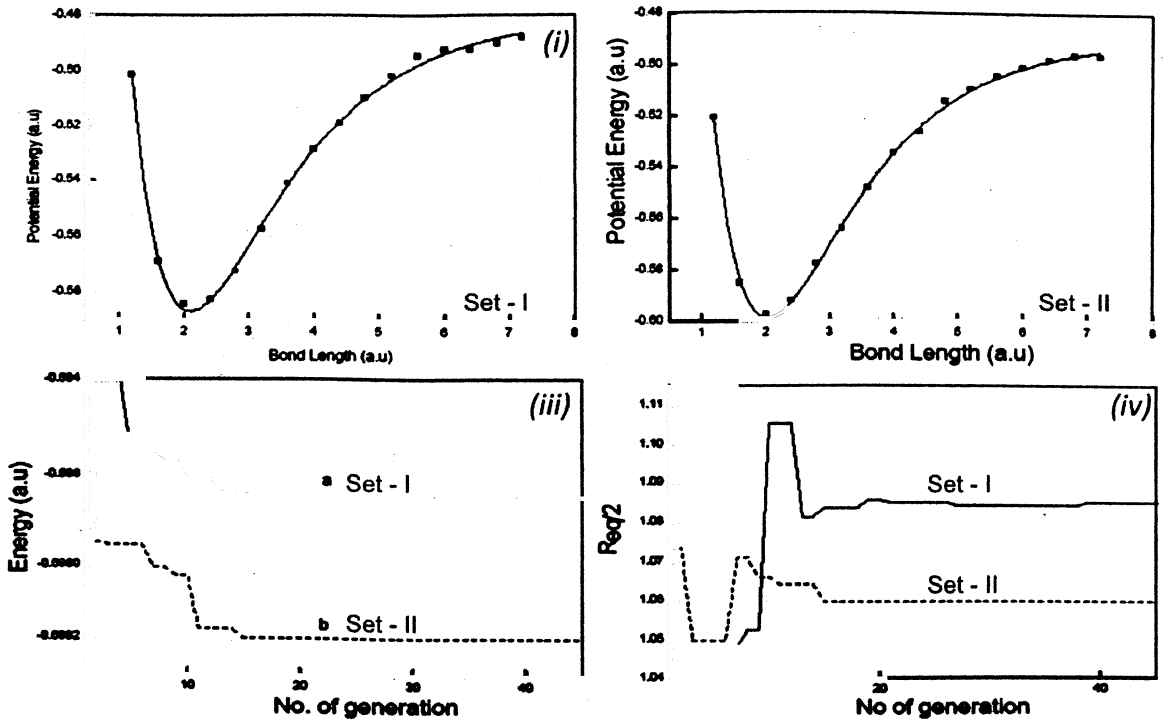


Fig. 3. (i) For the calculations of Set I (with 64×64×64 grid points) under Type I GA-based search energy-internuclear distance profile (computed by GA for the ground state of H^+ at different inter-nuclear distances) is fitted to a Morse curve, (ii) For the calculations of Set II (with 128×128×128 grid points) under Type I GA-based search energy-internuclear distance profile (computed by GA for the ground state of H^+ at different inter-nuclear distances) is fitted to a Morse curve, (iii) Under Type II GA-based search (simultaneous search for φ_e, R_e, V_e being carried out by GA) evolution of energy of the best string in the GA run is displayed for (a) Set I, and (b) Set II calculations, (iv) Under Type II GA-based search (simultaneous search for φ_e, R_e, V_e being carried out by GA) profile of the equilibrium bond length of the best string in the GA run is displayed for (a) Set I, and (b) Set II calculations.

stopped, the minimum energy (V_e) and equilibrium internuclear separation R_e were obtained in a single run. The evolution profile for the 'R' is shown in Fig. 3b. R_e in the second approach come out as 2.11 a.u. as opposed to the fixed -R approach where it turned out to be slightly smaller (2.03 a.u.). The computed values of R_e were, however, close (-0.599 a.u. in the first, and -0.598 a.u. in the second approach, both comparing very well with the literature value of -0.602 a.u.) (Type II).

(iii) The equilibrium probability distribution $|\Psi(x,y,z,R)|^2$ as $R\rightarrow\infty$ displayed the correct dissociation behavior into $H+H^+$ when the symmetry-constraint on the wave function was withdrawn[8]. The double-peaked distribution then passed over into a single peaked one as $R\rightarrow\infty$.

(c) Density based electronic structure calculation : Neutral polythiophene (PTs) is a semiconductor. When oxidized (hole-doped, in other words) to a critical level, it shows metallic conduction. The evolution of the electronic structure of doped PTs has therefore been at the focus of many theoretical investigations. A genetic algorithm-driven direct density method was recently proposed for the calculation of the electronic structures of doped and undoped PTs using a modified Su-Schrieffer-Heeger (SSH) Hamiltonian[9,10].

Let us consider a PT oligomer containing m-thiophene rings. The π MOs (Ψ_i) of the oligomers are constructed with the $4m$-carbon $2p_z$ and m-sulphur $3p_z$ orbitals (χ_ps)

$$\Psi_1 = \sum_{p=i}^{5m} C_{pi}\chi_p \qquad (21)$$

In matrix notation

$$\Psi(\Psi_1, \Psi_2, \ldots, \Psi_N) = \chi(\chi_1, \chi_2, \ldots \chi_{5m})C \qquad (22)$$

Let n be the number of π MOs that are doubly occupied

'at the particular doping level. The π-density matrix in the $C\text{-}2p_z$ and $S\text{-}3p_z$ basis is then given by

$$P = \sum_{i=1}^{n} c_i c_i^{+} = CC^{+} \tag{23}$$

The SSH Hamiltonian is defined in terms of the on-site energies (α_k), the hopping interaction energies between the nearest neighbor atoms (V_{kl}), and an elastic σ-compression energy $\left(\sigma = \frac{1}{2}\Sigma f(R_{ij})\right)$

$$H = H_\pi + H_\sigma$$

$$= \sum_k \alpha_k a_k^{+} a_k + \frac{1}{2}\left\{\sum_{k,l} V_{kl} a_k^{+} a_l + H.\right.$$

$$+ \frac{1}{2}\sum_{i \neq j} f(R_{ij}) \tag{24}$$

The $f(R_{ij})$ terms as well as the $V_{kl}s$ are parametrized. $E_G(R)$, the ground state energy for a given set of π MOs and given σ framework (R), is given by

$$E_G(R) = E_\pi(R) + E_\sigma(R) \tag{25}$$
$$= 2TrPH_\pi + E_\sigma(R)$$

The configuration R is defined by the set of nearest neighbor interatomic distances (R^0_{12}, R^0_{23},...) which define V_{kl} and $f(R_{ij})$ terms completely.

The GA density method proposed by Sharma *et al.*[11] works with a population of geometry string $S_1(R_{12}, R_{23}, ..., R_{ij}, ...)$, $S_2(R_{12}, R_{23}, ..., R_{ij}, ...)$, ..., $S_M(R_{12}, R_{23}, ..., R_{ij}, ...)$ which are allowed to evolve genetically by the action of selection, crossover and mutation operations. Each such string defines the π-framework of the oligomer, and therefore the π-electron Hamiltonian H_π completely. The genetic operators do not act upon the π-electron density matrices (P_k). The $P_k s$ are forced to co-evolve by the action of a unitary transformation $U(R)$ generated from the $H_\pi(R)$, which in turn is fixed by the geometry strings $S(R)$. Starting with the string $S_0(R)$ and the corresponding π-density matrix $P_0(R_0)$, we generate the density matrix $P_k(R_k)$ corresponding to the geometry coded by the string $S(R_k)$ as follows

$$S(R_0) \xrightarrow[\text{operations}]{\text{genetic}} S(R_k) \tag{26}$$

$$H(R_0) \xrightarrow{S(R_k)} H(R_k) \tag{27}$$

$$P_k(R_k) = U_k^{+}(R_k)\, P_0(R_0) U_k(R_k) \tag{28}$$
$$= e^{-i\lambda H_k(R_k)} P_0(R_0) e^{i\lambda H_k(R_k)}$$

where $P_0(R_0)$ has been obtained by diagonalizing $H(R_0)$, determining the C_0 matrix ($=\sum_i C_i C_i^{+}$) and thereby forming the P_0 matrix ($=C_0 C_0^{\dagger}$). The total energy of the oligomer is computed by adding the σ-framework energy ($E_\sigma(R_k) = \frac{1}{2}\Sigma f(R_{ij})_k$) to the π-electron energy ($E_p(R_k) = 2Tr\{P_k(R_k)H(R_k)\}$). The fitness function f_k for the oligomer corresponding to the geometry coded by $S_k(R_k)$ is defined as

$$f_k = e^{-\gamma(E(R_k)-E_L)^2} \tag{29}$$

which tends to unity as $E(R_k) \to E_L$ (E_L is the lower bound to the total energy of the oligomer, γ is a scalar at the choice of the user). The co-evolving density $P_k(R_k)$ automatically satisfies the constraints $2Tr(P_k) = N$ (no. of π-electrons) and $P^2_k(R_k) - P_k(R_k) = 0$, if $P_0(R_0)$ is idempotent and $2TrP_0(R_0) = N$. This is because the transformation matrix $U_k(R_k)$ is unitary. In practice, one must use a truncated expression for $P_k(R_k)$

$$P_k(R_k)) \approx P_0(R_0) + i\lambda[P_0(R_0), H_k] + \frac{i^2}{2!}\lambda^2$$
$$[P_0(R_0), H_k], H_k] \tag{30}$$

and that may compromise with the idempotency of $P_k(R_k)$ and the purity of the trace of $P_k(R_k)$. A small λ will usually ensure that the error remains negligible. The fitness evolution profiles for the undoped and doped polythiophene chain containing 100 thiophene rings are shown in Figs. 4a and 4b. The co-evolution strategy seems to work well. The strategy is perfectly generalizable for use with *ab-initio* methods.

Mutation only method : If we use a single string in the population ($n = 1$), there is no crossover process and so the evolution is guided by the mutation process only. Since the energy (or fitness) calculation is the most time consuming step, the multiple population-based strategy like the GA must be at-least n-fold more efficient in locating the fitness-maximal string than a single string based

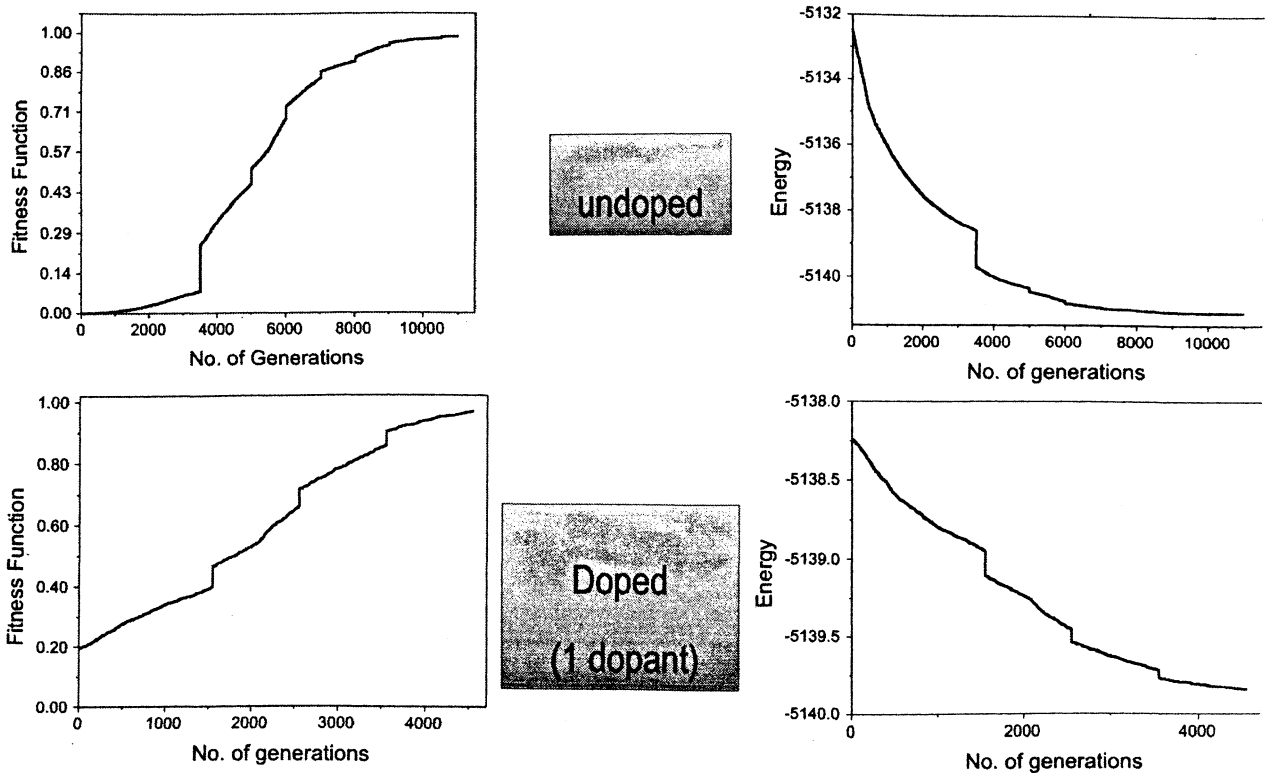


Fig. 4. Fitness evolution profiles of 100-ring polythiophene chain for a (a) neutral polythiophene, (b) doped polythiophene. Energy evolution profiles for a (a) neutral polythiophene, (b) doped polythiophene.

strategy. The linear scalability, however, is not guaranteed for most of the fitness landscapes. We therefore thought it worthwhile to try to develop a single string based mutation only strategy of fitness maximization. The mutation process defined on floating point strings has two flexible parameters viz. mutation probability (P_m) and mutation intensity (Δ_m). We have proposed a mutation only strategy in which both P_m and m are adaptively determined by the algorithm[10,12-14]. Moreover, we insist on strong 'elitism' in that a mutation is 'accepted' only when it does not lower the fitness. The resulting algorithm has been called Adaptive Random Mutation Hill Climbing (ARMHC) method. We demonstrate its working principles and workability with reference to determining the global minimum potential energy configurations of 3-D Coulomb clusters under harmonic confinement.

3-D Coulomb clusters

There had been recent excitement in the field of colloid chemistry in the context of the question do colloids would crystallize? Colloids are made up of similarly charged particles of nanometer size (there are enough counter ions, of course, for maintaining the charge neutrality) and are akin to the confined 3-D Coulomb clusters[13]. The algorithm that works for locating the global minima in Coulomb clusters may be expected to work on the PES of colloids as well. The ARMHC strategy with reference to Coulomb clusters is detailed in what follows.

In a system of N identical point charges (q_i) confined by a parabolic trap, the potential energy $V(x_1^0, y_1^0, z_1^0, \ldots, x_N^0, y_N^0, z_N^0)$ is given by

$$V(x_1^0, y_1^0, z_1^0, \ldots, x_N^0, y_N^0, z_N^0) = \sum_{i=1}^{N-1}\sum_{j>i}^{N} \frac{q_i q_j}{r_{ij}} + \frac{1}{2}k\sum_{i=1}^{N}(x_i^2 + y_i^2 + z \quad (31)$$

The positional coordinates of charges are randomly chosen to start with and encoded by the string S_0 where

$$S_0 = S(x_1^0, y_1^0, z_1^0, \ldots, x_N^0, y_N^0, z_N^0)$$

$$\equiv S(\xi_1^0, \xi_2^0, \xi_3^0, \ldots, \xi^0_{3N-1}, \xi^0_{3N}) \quad (32)$$

The initial potential energy is therefore given by

$$V_0 = \sum_{i=1}^{N-1}\sum_{j>i}^{N} \frac{q_i q_j}{r_{ij}^0} + \frac{1}{2}k\sum_{i=1}^{N}(r_i^0)^2 \quad (33)$$

where

$$r_{ij}^0 = \sqrt{(x_i^0 - x_j^0)^2 + (y_i^0 - y_j^0)^2 + (z_i} \quad (34)$$

and

$$r_i^0 = \sqrt{(x_i^0)^2 + (y_i^0)^2 + (z_i^0)^2}$$

Since V is positive the fitness of the string S_0 can be defined as the reciprocal of V_0, i.e.

$$f_0 = \frac{1}{V_0} \quad (35)$$

One or more of the $3N$ coordinates ($\xi_1^0, \xi_2^0, \xi_3^0, \ldots, \xi^0_{3N-1}, \xi^0_{3N}$) are randomly picked up with a probability P_m and are allowed to mutate with an intensity Δ_m, generating a new configuration of charges. In general, the j-th configuration created by mutation of the (j-1)-th configuration of charges can be written as

$$\xi_k^j = \xi_k^{j-1} + (-1)^l \Delta_m r(0,1) \quad (36)$$

where $r(0,1)$ is a normally distributed random number in the range 0 to 1 and 'l' is a random integer. Both P_m and Δ_m are adaptively determined by the algorithm on the basis of continuous evaluation of the performance of the search for the global minimum[13].

If V_j is the potential energy of the j-th configuration of charges, the fitness is, as already stated in eq. (35)

$$f_j = \frac{1}{V_j} \quad (37)$$

if $f_j > f_{j-1}$, the j-th configuration is accepted as the new configuration and is used as the starting configuration for the next mutation; otherwise, the previous configuration [(j-1)-th] is retained and a new mutationvis generated. The process is continued till the fitness is maximized and system stops evolving any further.

Fig. 5 shows the fitness evolution profile in a 3-D Coulomb cluster of 300 charges confined by an isotropic and harmonic trap ($K_x = K_x = K_x = 1.0$). For comparison, the fitness evolution profile in a GA-based search (population size 30) is also displayed in the same figure. In this particular application, as can be clearly seen, ARMHC perform better than the GA, but no general conclusion about the relative performance of the two methods should be made right now.

Fig. 6 displays the structural evolution in a 50-charge system confined by a harmonic trap which is anisotropic

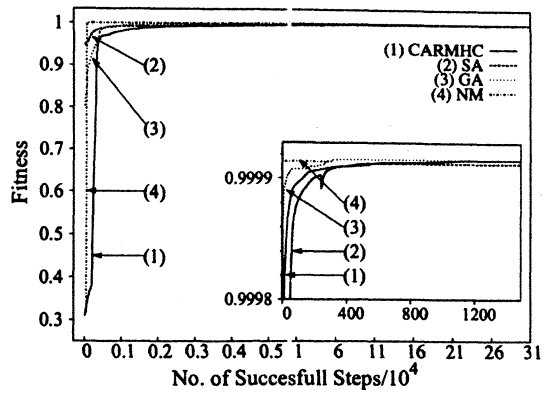


Fig. 5. Comparison of the fitness evolution profile obtained for CARMHC algorithm with those of SA, GA, and NM for a cluster of N=300 unit positive charges, confined in a harmonic trap.

($K_x = K_y \neq K_z$). The structural evolution is monitored as a function of K_z. It appears that the structural phase transition that takes place is continuous. However, a much closer examination of the process is needed to understand the nature of the transition completely.

The method proposed should be applicable to the study of colloid crystals, if a suitable representation of the PES on which colloid particles move, is available. Needless to

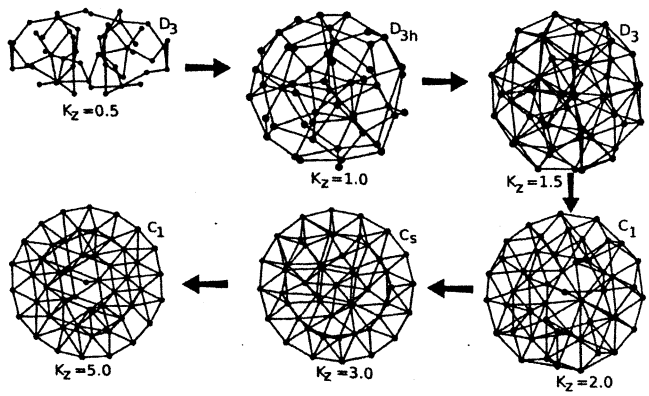


Fig. 6. Structural evolution for a system consisting of 50 charge particles with varying confinement strength along Z-axis (K_z). The point symmetry of each structure are also displayed.

mention, colloid particles being nm sized, point charge approximation cannot be used. We hope to address the problem elsewhere.

Acknowledgement

SPB wishes to thank the DAE, Government of India for the award of Raja Ramanna Fellowship, the authorities of IITB for hosting the scheme. KS thanks the CSIR, Government of India, New Delhi for the award of Senior Research Fellowship.

References

1. J. H. Holland, "Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence", MIT Press, Cambridge, MA, USA, 1992.
2. D. E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st ed., 1989.
3. Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs, Artificial intelligence", Springer, 1996.
4. R. Saha, P. Chaudhury and S. P. Bhattacharyya, *Physics Letters (A)*, 2001, **291**, 397.
5. P. Chaudhury and S. P. Bhattacharyya, *Chemical Physics Letters*, 1998, **296**.
6. P. Chaudhury and S. P. Bhattacharyya, *International Journal of Quantum Chemistry*, 1999, **74**, 153.
7. R. Saha and S. P. Bhattacharyya, *Journal of Theoretical and Computational Chemistry*, 2004, **03**, 325.
8. R. Saha and S. P. Bhattacharyya, *International Journal of Modern Physics (C)*, 2007, **18**, 163.
9. S. Nandy, P. Chaudhury, R. Sharma and S. P. Bhattacharyya, *Journal of Theoretical and Computational Chemistry* 2008, **07**, 977,
10. K. Sarkar, R. Sharma, and S. P. Bhattacharyya, *Journal of Chemical Theory and Computation*, 2010, **6**, 718.
11. R. Sharma, S. Nandy, P. Chaudhury and S. P. Bhattacharyya, *Materials and Manufacturing Processes*, 2011, **26**, 354.
12. K. Sarkar, R. Sharma and S. P. Bhattacharyya, *International Journal of Quantum Chemistry*, 2012, **112**, 1547.
13. K. Sarkar and S. P. Bhattacharyya, *AIP Conference Proceedings*, 2013, **1512**, 162.
14. R. Sharma and S. P. Bhattacharyya, Direct search for wave operator by a genetic algorithm (ga) : Route to few ei genvalues of a hamiltonian, in IEEE Congress on Evolutionary Computation, 2007, pp. 3812-3817.