MATCH
Communications in Mathematical and in Computer Chemistry

MATCH Commun. Math. Comput. Chem. **94** (2025) 77–93
ISSN: 0340–6253
doi: 10.46793/match.94-1.077A

Generating Molecules with Specific Boiling Points and Melting Points

Steven A. Alexander[a,*]

[a]*Department of Physics Southwestern University Georgetown, TX 78626*

alexands@southwestern.edu

(Received May 16, 2024)

Abstract

We examine a simple algorithm that uses simulated annealing to find molecules with a specific boiling point and melting point. For testing purposes, we consider molecules that contain only carbon, oxygen, nitrogen and hydrogen atoms. We represent these molecules as SMILES strings and use seven distinct operators to modify these strings.

1 Introduction

Most Ab Initio and semiempirical chemical methods can easily calculate a variety of molecular properties. Solving the inverse problem, finding a given molecule that has a particular property, is difficult because the solution space is vast and the variables are discrete rather than continuous. Any algorithm developed to find molecules with specific properties requires a mechanism for describing the molecule's structure, a process to change that structure and an optimization program to guide those changes toward the property of interest. Many methods arrange a set of chemical building blocks (atoms, functional groups, etc.) using either combinatorial methods [6, 10, 14] or genetic algorithms [3, 20, 33]. Because these building blocks

*Corresponding author.

describe a molecule as a simple string (e.g., ABC...), the program can easily restrict the generated compounds to specific chemical families such as acids, alcohols, or phenols.

Kvasnicka and Pospichal developed an alternative to the building block approach [19]. They used simulated annealing to search for cyclic and acyclic hydrocarbons with simple properties. Their paper used a tree structure to represent these molecules and described operations on this structure in the language of molecular graph theory. By altering the branches of these trees, Kvasnicka and Pospichal generated new molecules that became the inputs to their simulated annealing algorithm.

Several years later, Alexander used simulated annealing to calculate a series of neutral acyclic hydrocarbons with specific magnetic susceptibilities [2]. This annealing algorithm searched for the desired molecule by adding or deleting individual carbon atoms on a square lattice; an adjacency matrix recorded which atoms were connected and with what type of bond (single, double or triple). Although the results in this paper were encouraging, when we tried to generalize our method to accommodate molecules with more than one type of atom or molecules with rings, our solutions were almost always worse than those from the acyclic hydrocarbon calculation.

In this paper, we use a SMILES string [34] to represent a molecule's structure and seven mutation operators to modify a SMILES string from one form into another. A brief description of this molecular representation method is presented in Section 2. In Section 3 we outline how simulated annealing can be used to generate a set of molecules with specific properties. To illustrate the performance of this algorithm, we use it to find a molecule that has a specified boiling point and melting point. For testing purposes, we consider only molecules that contain carbon, oxygen, nitrogen, and hydrogen atoms. The rings in our molecules are also limited to those with 4, 5 or 6 atoms. Finally, in Section 4, we take a closer look at the convergence of our algorithm and consider several possible improvements.

2 Using SMILES to produce random molecules

SMILES is an acronym for Simplified Molecular Input Line Entry Specification. It uses a short ASCII string without spaces to describe a molecule's structure. The original version of SMILES was developed by Arthur Weininger and David Weininger in the late 1980s [34]. Since then, Weininger and others have modified SMILES to describe a number of additional features, such as chirality, reactions and disconnected structures [35, 36].

Four simple rules define a valid SMILES string:
* Atoms are described by their standard atomic symbol. Each symbol is normally enclosed in square brackets, such as [Au] for gold, however, many of the atoms found in organic molecules (such as B, C, N, O, P, S and F) are written without brackets. Most SMILES strings omit all hydrogen atoms. The implicit number of hydrogen atoms attached to other atoms is the difference between the atom's valence and the number of bonds assigned to the atom.
* Single, double and triple bonds are represented by the symbols '-', '=' and '#' respectively. The atoms connected by these bonds are indicated by their adjacency. In most versions of SMILES, single bonds are omitted from the string, but for convenience, we explicitly show all bonds.
* Branching is specified by placing the symbols for the atoms and bonds in this subchain between parentheses. These parentheses are placed directly after the symbol for the atom on the main sequence to which it is connected.
* Rings are represented by breaking a single bond in each ring and then designating the two atoms connected by this bond with a digit immediately following the symbol for the atoms.

Part of the power of SMILES is that each molecule has a unique SMILES string. Although this string contains the same information as might be found in an extended connection table, a SMILES string can also be thought of as a language - albeit one with a simple vocabulary (atom and bond symbols) and a few grammar rules. If we want to transform an initial molecule into one that has a specific property, seven operators are

needed to convert one SMILES string into a new SMILES string:
* Pick a random bond and change it into a different type of bond (e.g. C=C-N-O → C-C-N-O)
* Add a random atom with a random bond to a random location in the SMILES string (e.g. C=C-N-O → C=C-O-N-O)
* Add a random atom with a random bond as a new branch to a random location in the SMILES string (e.g. C=C-N-O → C=C-N(-O)-O)
* Delete a random atom and its connecting bond (e.g. C=C-N-O → C=C-O)
* Pick a random atom in the SMILES string and change it into a different type of atom (e.g. C=C-N-O → C=C-N-C)
* Pick two random atoms in the SMILES string and connect them with a ring (e.g. C=C-N-O → C1=C-N-O1)
* Delete a ring (e.g. C1=C-N-O1 → C=C-N-C)

3 Generating molecules with a specific boiling point and melting point

Several numerical methods can search for the global minimum of a complicated multidimensional function. Such functions often have several local minima, making it difficult to determine if a particular solution is in a local or a global minimum. One well-known optimization technique, simulated annealing, has been successfully applied to problems with both discrete and continuous variables [4, 9, 17, 32]. This method attempts to avoid the problem of getting stuck in a local minimum by occasionally accepting steps that yield worse solutions. Starting from some initial point, $x_{initial}$, and its value at that point, $f(x_{initial})$, simulated annealing generates a new point in the multidimensional space, x_{new}, and calculates its value at that point, $f(x_{new})$. If $f(x_{initial}) > f(x_{new})$ this step is accepted and x_{new} becomes the starting point for the next step. If $f(x_{initial}) < f(x_{new})$, an acceptance function determines whether x_{new} is accepted or rejected. Although other acceptance functions have been described in the literature [29,31], the most popular choice is the Metropolis function [28]

$$A = min(R, exp([f(x_{new}) - f(x_{initial})]/T)) \qquad (1)$$

Here T is a parameter known as the temperature and R is a random number between 0.0 and 1.0.

In most simulated annealing calculations, the temperature starts at a high value. This allows a sequence of steps to efficiently sample the whole parameter space since most steps are accepted. After a certain number of steps, the temperature is decreased. Several cooling schedules have been examined in the literature [13,27,30] but a common choice is the simple geometric relation

$$T_{k+1} = CT_k \qquad (2)$$

where C is a constant between zero and one. As the temperature is reduced, some steps may collect in a local minimum. If the temperature decreases slowly, these steps can escape from the local minimum and eventually reach the global minimum. Most descriptions of this method claim that the rate at which the temperature decreases depends on the complexity of the fitness function and on the number of steps it takes the system to reach an equilibrium at each temperature.

To illustrate how simulated annealing can find molecules with a specific boiling point (BP) and melting point (MP), we set our fitness function to

$$f(x_i) = \sqrt{(MP_i - MP_0)^2 + (BP_i - BP_0)^2} \qquad (3)$$

Here MP$_i$ and BP$_i$ are the computed values for a given molecule and MP$_0$ and BP$_0$ are the desired values. In the literature, several methods use group contributions to estimate boiling and melting points [7,16,18, 26]. The group contribution approach defines a small number of simple functional groups and assigns a numerical value to each group for each property. To calculate a particular molecule's boiling point or melting point, we need to break that molecule into its different functional groups and then sums the values of each group. Because of its simplicity, we will use the version proposed by Joback and Reid [16]. Since some of Joback

Table 1. Molecules generated with $MP_0 = 147.67°C$, $BP_0 = 336.51°C$, an initial temperature of 10, ten temperature reductions and 1000 steps at each temperature.

Initial Molecule	f($x_{initial}$)	Final Molecule	f(x_{final})
C(=C)-C(-N)(-C)-O	173.033	C(-C)(=C=C)(-C(-C))	0.00000225
C(=C-N)-C-C	71.9217	C(=C(-C(-C))-C)=C	0.00000225
C-C=C-N-O	114.510	C(-C)(=C=C)(-C(-C))	0.00000225
C(-C)-C(-N)-O	141.752	C(-C)(=C=C)(-C(-C)))	0.00000225
C(-N-C=C-C)-C	55.4246	C(-C(-C-C)(=C=C))	0.00000225
C1-N-C(=C1)-O-C	152.712	C(-C)(=C=C)(-C(-C))	0.00000225
C(=C1)-C-N-C(=C1)-O	238.900	C(=C=C(-C(-C))(-C))	0.00000225
C1-C2-C-C(=O)-C2-C1	139.558	C(-C(=C(=C))(-C-C))	0.00000225
C1-N=C-O-C(-O)-N1	326.376	N(=C=O)-C-C	1.835
C1-C-C2=C-C-C12	66.5166	C(-C)-C(-C)(=C(=C))	0.00000225

and Reid's contributions distinguish between an atom in a ring and one that is not, we also need a method to identify how many rings a molecule has and which atoms are in a ring. Several numerical techniques can provide this information from the molecule's adjacency matrix [11, 15, 25, 37]. Because of its simplicity, we use the algorithm described by Lau [24].

We begin our calculations with an initial temperature of T=10 and perform ten temperature reductions using a value of C = 0.75 in Eqn. 2. At each temperature, we execute a fixed number of steps. At the beginning of each step, we use each of the mutation operators described in the previous section to modify the current string. Of course, not all operators will produce a valid SMILES string (a change in the bond type could cause the number of bonds attached to an atom to exceed that atom's valence). If an operator creates an invalid string, we discard that string. The string with the best fitness function is then used in Eqn. 1 to determine whether the original string or this new string will become the basis for the next step. Rather than performing this simulated annealing algorithm on only one molecule, we examine ten randomly generated initial molecules simultaneously. The results from these different molecules will allow us to explore the system's convergence. In an ideal run, all the molecules should find the global minimum.

The first example we examined has a target melting point of 147.67°C and a boiling point of 336.51°C. We chose this combination because there

Table 2. Molecules generated with $MP_0 = 270.57°C$, $BP_0 = 362.65°C$, an initial temperature of 10, ten temperature reductions and 1000 steps at each temperature.

Initial Molecule	f($x_{initial}$)	Final Molecule	f(x_{final})
C(=C)-C(-N)(-C)-O	86.847	C(=C1)-O(-N-C1)	0.00000279
C(=C-N)-C-C	58.261	C(=C-N-C1)-O1	0.00000279
C-C=C-N-O	64.980	C(=O)=C-N-N	0.771
C(-C)-C(-N)-O	72.054	C(=C1)-C(-N-O1)	0.00000279
C(-N-C=C-C)-C	77.573	C(-N-C-O1)=C1	0.00000279
C1-N-C(=C1)-O-C	29.848	O1-N-C(=C(-C)1)	0.00000279
C(=C1)-C-N-C(=C1)-O	136.694	C#C-C(-C(-O-C)(-C(#C)))	1.522
C1-C2-C-C(=O)-C2-C1	56.154	N(-O(-C1))-C=C1	0.00000279
C1-N=C-O-C(-O)-N1	211.380	N(-O(-C(=C)1))(-C1)	0.00000279
C1-C-C2=C-C-C12	60.402	C1-N-C=C-O1	0.00000279

is an acyclic molecule with these values using Joback and Reid's model. As Table 1 shows, nine of the ten initial molecules quickly reproduced this target molecule using only 1000 steps at each temperature. Although the outlier in this calculation significantly improved its fitness function, it has become locked in a local minimum that can't delete the nitrogen and oxygen atoms.

The second example we examined has a target melting point of 270.57°C and a boiling point of 362.65°C. We chose this combination because there is a cyclic molecule with these values using Joback and Reid's model. As Table 2 shows, eight of the ten initial molecules quickly reproduced this target molecule using only 1000 steps at each temperature. In both cases, the outliers in this calculation failed to produce a molecule with a ring.

Our last example has a target melting point of 200.0°C and boiling point of 400.0°C. We chose this combination because we knew of no molecule that satisfied both requirements in Joback and Reid's model. As a result, our optimization algorithm has to balance these two demands as best as possible. Using 1000 steps we see in Table 3 that the fitness function of our 10 initial molecules undergoes a substantial improvement, but they have not converged to a final molecule. Instead, these final values exhibit a large range (0.443-1.243) and a relatively large variation of $\sigma = 0.336$. The most likely reason for this behavior is that the number of steps

Table 3. Molecules generated with MP_0 = 200.0°C, BP_0 = 400.0°C, ten temperature reductions and 1000 steps at each temperature.

Initial Molecule	$f(x_{initial})$	Final Molecule	$f(x_{final})$
C(=C)-C(-N)(-C)-O	93.072	C=C(-N-O(-C))-N(=O)	0.907
C(=C-N)-C-C	34.725	C(-C(-C-C(=C=O)(-C(-C))))-C(-C)	1.070
C-C=C-N-O	34.670	C(=C(-C-N=C-C(#C)))	0.477
C(-C)-C(-N)-O	61.347	C(=C=C-C)=C(=C=C(-C))	0.496
C(-N-C=C-C)-C	32.589	O(-O(-C(=C)(-C(=C))))	1.243
C1-N-C(=C1)-O-C	95.311	O(-C(=C)(-O(-C(=C))))	1.243
C(=C1)-C-N-C(=C1)-O	159.793	O(-C(-C)(-C(=C=O)(-C)))	0.443
C1-C2-C-C(=O)-C2-C1	63.575	C(=C)(-C(=C)-O-O)	1.243
C1-N=C-O-C(-O)-N1	250.433	O(-C(=C)-O(-C=C))	1.243
C1-C-C2=C-C-C12	42.298	O(-O(-C(=C)-C=C))	1.243

at each temperature is too small to sample the essential regions of the parameter space. Table 4 examines what happens when we increase the number of steps at each temperature from 1000 to 5000. Of the 10 initial molecules, eight have a final fitness value of 0.477, one has a higher value of 0.907 and one has a value of 0.123 (which is lower than any of the values in Table 3). This combination has a smaller range of values (0.123-0.907) and a smaller variation of σ = 0.185 than our earlier calculation. The arrangement of the final molecules into three groups suggests that the fitness landscape for this combination of melting point and boiling point has several shallow minima. The fact that we had to use 5000 steps at each temperature to get this level of convergence supports this idea. Increasing the number of steps at each temperature step to 10,000 didn't improve our results.

4 Computational efficiency

In the previous section, we computed the first two examples using an initial temperature of 10, ten temperature decreases and 1000 steps at each temperature. We generated seven new strings at each step, one from each mutation operator. Thus, a complete run for a single molecule required us to compute the fitness function 7*10*1000 = 70,000 times. Fortunately, our program can evaluate Joback and Reid's model quickly; a complete calculation with ten initial molecules took about a minute on a desktop

Table 4. Molecules generated with MP_0 = 200.0°C, BP_0 = 400.0°C, an initial temperature of 10, ten temperature reductions and 5000 steps at each temperature.

Initial Molecule	$f(x_{initial})$	Final Molecule	$f(x_{final})$
C(=C)-C(-N)(-C)-O	93.072	C(#C-C-C(=C(-N(=C))))	0.477
C(=C-N)-C-C	34.725	C(#C-C-C(=C(-N(=C))))	0.477
C-C=C-N-O	34.670	N(-C-C#C)(=C-C(=C))	0.477
C(-C)-C(-N)-O	61.347	C(=C)(-C(-C=N-C#C))	0.477
C(-N-C=C-C)-C	32.589	C(=N-C#C)(-C(-C(=C)))	0.477
C1-N-C(=C1)-O-C	95.311	C(#C(-C(-C=C(-N(=C)))))	0.477
C(=C1)-C-N-C(=C1)-O	159.793	C(-C=C)(-C(=N(-C(#C))))	0.477
C1-C2-C-C(=O)-C2-C1	63.575	N(=C-C-C#C)(-C=C)	0.477
C1-N=C-O-C(-O)-N1	250.433	N(=O)(-C(=C)(-O(-N-C)))	0.907
C1-C-C2=C-C-C12	42.298	C(=C-C(=C(-N(-C)-C=C(-C))))	0.123

computer. Although almost all of our initial molecules eventually found the global minimum in these two examples, the fact that a few did not suggests that there are opportunities for further improvement.

As a starting point for studying the behavior of our algorithm, we selected initial molecule #1 in Table 1. If we plot the fitness functions of the current molecule and the best molecule during the first 200 steps, Figure 1 shows that the fitness function of the current molecule decreases rapidly during the first few steps. This drop occurs because the mutation operators are producing changes that move the current molecule towards the target molecule with the desired properties; many of these changes become new best molecules. After this brief initial stage, however, it takes longer for the mutation operators to produce a new best molecule. This is because after Eqn. 1 chooses a molecule with a worse fitness function, the fitness function often returns to the current low after a few steps.

Another way we can study the convergence of our simulated annealing algorithm after this initial phase is to fix the temperature at T=10 and accumulate three data points over groups of 200 steps: the number of new lows found, the number of worse molecules selected by Eqn. 1 and the number of invalid molecules generated by the mutation operators. This information is presented in Table 5. As expected, the first group has the most new lows. The number of worse molecules selected by Eqn. 1 is also highest (about 22%) in the first two groups. In later groups, the convergence slows because the mutation operators are generating many invalid

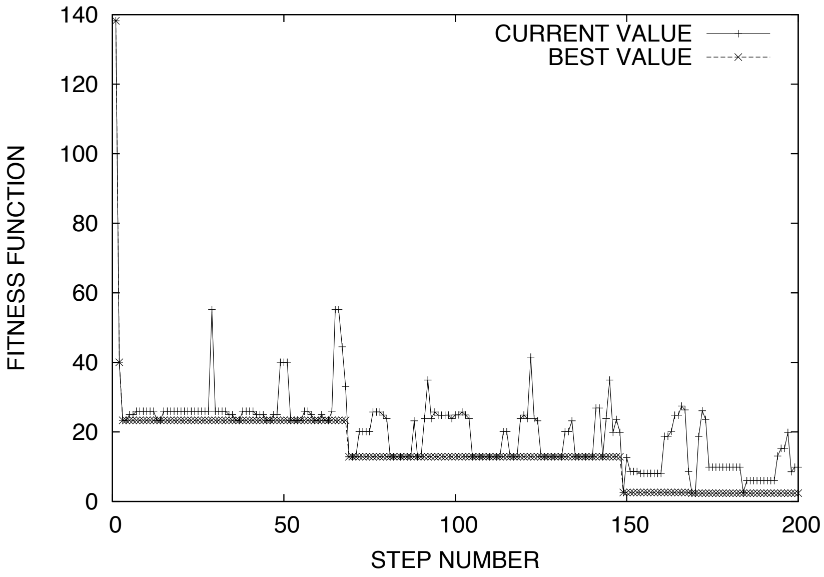

Figure 1. Convergence of initial molecule #1 with MP_0 = 147.67°C, BP_0 = 336.51°C and T=10.

Table 5. Convergence of initial molecule #1 with MP_0 = 147.67°C, BP_0 = 336.51°C and groups of 200 steps at T=10.

Group Number	1	2	3	4	5	6	7	8	9	10
New lows found	6	0	1	0	0	1	0	0	0	0
Worse molecules selected	42	45	28	26	29	33	27	18	17	26
Invalid molecules generated	731	736	916	946	958	840	964	943	1001	926

molecules. The percentage of these invalid molecules ranges from 52% in the first group to 71% in the ninth group. Because the target in this example is an acyclic system, these numbers are artificially high since neither of the ring mutation operators (#6 and #7) contribute to the final solution. The fact that there are fewer invalid choices at the beginning of this run than at the end suggests that randomly selecting atoms and bonds in the mutation operators becomes less efficient as the current molecule approaches a minimum. Surprisingly, the calculation used to produce Table 5 located the global minimum during the sixth group without performing any reductions in temperature. In a 1990 paper, Connolly proposed that

Table 6. Molecules generated with MP_0 = 147.67°C, BP_0 = 336.51°C and a different number of steps at T=10.

Initial Molecule	$f(x_{initial})$	$f(x_{final})$ 1000 steps	$f(x_{final})$ 2000 steps
C(=C)-C(-N)(-C)-O	173.033	1.088	0.00000225
C(=C-N)-C-C	71.9217	1.835	0.00000225
C-C=C-N-O	114.510	1.088	0.00000225
C(-C)-C(-N)-O	141.752	1.088	0.00000225
C(-N-C=C-C)-C	55.4246	0.00000225	0.00000225
C1-N-C(=C1)-O-C	152.712	1.088	0.00000225
C(=C1)-C-N-C(=C1)-O	238.900	1.088	0.00000225
C1-C2-C-C(=O)-C2-C1	139.558	1.835	0.00000225
C1-N=C-O-C(-O)-N1	326.376	1.835	0.00000225
C1-C-C2=C-C-C12	66.5166	2.389	0.00000225

Table 7. Molecules generated with MP_0 = 270.57°C, BP_0 = 362.65°C and a different number of steps at two fixed temperatures.

Initial Molecule	$f(x_{initial})$	T=10 $f(x_{final})$ 1000 steps	T=10 $f(x_{final})$ 2000 steps	T=20 $f(x_{final})$ 1000 steps	T=20 $f(x_{final})$ 2000 steps
C(=C)-C(-N)(-C)-O	86.847	0.00000279	0.00000279	0.00000279	0.00000279
C(=C-N)-C-C	58.261	0.00000279	0.00000279	0.00000279	0.00000279
C-C=C-N-O	64.980	0.771	0.771	0.747	0.00000279
C(-C)-C(-N)-O	72.054	0.00000279	0.00000279	0.00000279	0.00000279
C(-N-C=C-C)-C	77.573	0.00000279	0.00000279	0.00000279	0.00000279
C1-N-C(=C1)-O-C	29.848	0.00000279	0.00000279	0.00000279	0.00000279
C(=C1)-C-N-C(=C1)-O	136.694	4.243	4.243	0.00000279	0.00000279
C1-C2-C-C(=O)-C2-C1	56.154	0.771	0.771	0.771	0.00000279
C1-N=C-O-C(-O)-N1	211.380	5.409	0.747	0.00000279	0.00000279
C1-C-C2=C-C-C12	60.402	0.00000279	0.00000279	0.00000279	0.00000279

running an annealing algorithm at a constant temperature could be helpful in some types of optimization problems [5]. The values in Table 4 support the usefulness of this idea, so we decided to run this type of calculation for all of the initial molecules in Table 1. As Table 5 shows, all of our initial molecules found the global minimum. Not only is this an improvement in the quality of our earlier results but this new calculation is also more computationally efficient. Each molecule only requires 7*1*2000=14,000 function evaluations, which is a substantial reduction from our earlier value of 70,000.

If we apply this modification to the second example, Table 6 shows that

Table 8. Molecules generated with MP_0 = 200.0°C, BP_0 = 400.0°C, using three groups of 5000 steps and three temperatures (T=5, 10 and 20) in each group.

Initial Molecule	$f(x_{initial})$	Final Molecule	$f(x_{final})$
C(=C)-C(-N)(-C)-O	93.072	C(-C=C)(=C(-N(-C)(-C(=C-C))))	0.123
C(=C-N)-C-C	34.725	C(=C(-C(=C)))(-C=C(-N(-C)(-C)))	0.123
C-C=C-N-O	34.670	C(=C-C)(-N(-C)-C(=C-C=C))	0.123
C(-C)-C(-N)-O	61.347	C(=C(-C))(-N(-C)-C(=C(-C=C)))	0.123
C(-N-C=C-C)-C	32.589	C(-C(-C)(-C(=C(=O))(-O)))	0.443
C1-N-C(=C1)-O-C	95.311	O(-C(-C(-C)(-C))=C(=O))	0.443
C(=C1)-C-N-C(=C1)-O	159.793	C(-C)(=C=O)(-C(-C)-O)	0.443
C1-C2-C-C(=O)-C2-C1	63.575	C(-C(-C)(-C(=C(=O))(-O)))	0.443
C1-N=C-O-C(-O)-N1	250.433	C(-C(-O)(-C(-C)(=C=O)))	0.443
C1-C-C2=C-C-C12	42.298	C(-C(-C)(-C(=C(=O))(-O)))	0.443

four out of ten molecules were not able to reach the global minimum when we kept the temperature at T=10. Raising the temperature to T=20, however, allows our new annealing algorithm to explore the solution space more easily, enabling all the initial molecules to find the global solution in only 2000 steps.

In contrast to the first two examples, performing a simulated annealing calculation at a constant temperature did not improve our results for the third example. Neither a run of 5000 steps at T=10 or T=20 produced better values nor did increasing the number of steps to 10,000. Several studies have shown that the optimum temperature in a simulated annealing calculation depends on the type of optimization problem [1, 22]. It should be high enough to allow easy movement but not so low that the system is frozen. A novel alternative to searching for this optimum temperature was first explored by Delamarre and Virot in 1998 [8]. They performed a short run on the same initial point at multiple temperatures. The best solution from this run became the starting point of the next iteration. Using this idea, Graffigne obtained good convergence for several difficult problems [12]. We explored this idea by performing a simultaneous annealing run of 5000 steps at three temperatures (T=5, 10 and 20). After this calculation finished, we picked the molecule with the lowest fitness value and then repeated this process two more times. As Table 7 illustrates, each initial molecule shows significant improvement compared to the results in Table 3. Four of the ten initial molecules were able to locate what we believe is the global minimum, 0.123. The rest have settled into a new local minimum of 0.443, which is better than the others in Table 3. One noticeable difference between the two minima in Table 7 is the number of atoms. The global minimum is described by a molecule with eight carbon atoms and one nitrogen atom. In contrast, the higher minimum is described by a molecule with five carbon atoms and two oxygen atoms. This difference suggests that these two minima are widely separated in the solution space of this problem and could mean that our simultaneous annealing program still has trouble sampling all of the solution space. Despite this problem, this modified algorithm is much more computationally efficient than our original program. The calculations described in Table 3 took 7*10*5000=350,000 function evaluations, but this simultaneous annealing calculation took only 7*3*5000=105,000.

5 Conclusions

Two recent studies have combined SMILES strings and multidimensional optimization routines to search for molecules with specific properties. Lameijer's Ph.D. thesis [23] employed an evolutionary algorithm to generate molecules with "good" chemical parameters (such as molecular weight, the number of rotatable bonds or the number of aromatic substituents). He found that his program quickly produced many molecules that were worthy of further investigation. In contrast, Kwan and Lee examined an evolutionary algorithm that optimized targets with a modified drug-likeness score [21]. Their algorithm included both mutation operators (although curiously only those that modify atoms but not bonds) and crossover operators. The fact that both these studies could generate molecules of interest using SMILES strings and optimization routines gives us confidence that our approach to the inverse problem is promising.

In this paper, we have shown that a simple simulated annealing algorithm can find cyclic and acyclic molecules with specific boiling and melting points as described by the method described by Joback and Reid [16]. When we run this program using multiple initial molecules, the results of these calculations provide useful information about calculation's convergence and the solution space's structure. Although our tests considered only molecules with carbon, oxygen, nitrogen, and hydrogen atoms, we can easily extend our algorithm to SMILES strings containing any atom and or ring size.

One advantage of Joback and Reid's model is that it requires very little time to compute the fitness function. If we had used a semiempirical, Ab Initio, density functional theory or molecular mechanics program to evaluate the fitness function, each iteration would have required much more time. The computational cost of such a calculation will be a significant challenge if we want to apply our algorithm to more realistic problems. Since Metropolis et al. first described this algorithm [28], researchers have developed several variants that solve specific problems more efficiently. This paper examined some of these modifications to see whether they could help find molecules with specific properties. While our results show that some improvement is easily obtainable, the problems with the third example (not all the initial molecules ended up in the global minimum and the large number of steps required to reach this result) suggest that further modifications to our algorithm are still needed.

***Acknowledgment*:** I want to thank Dr. Walt Potter for his friendship and many valuable discussions. I also want to thank Southwestern University for funding the 2007 SURGE program during which the early part of this research was conducted with the help of Stephen Foster, Nathan Lindsay, Tommy Rogers and Carl West.

References

[1] E. H. L Aarts, J. H. M. Horst, *Simulated Annealing and Boltzmann Machines*, Wiley, Chichester, 1989.

[2] S. A. Alexander, Generating molecules with specific properties: acyclic hydrocarbons, *MATCH Commun. Math. Comput. Chem.* **55** (2006) 305–314.

[3] N. Brown, B. McKay, J. Gasteiger, The de novo design of median molecules within a property range of interest, *J. Comput. Aided Mol. Design* **18** (2004) 761–771.

[4] V. Cerny, Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm, *J. Optim. Theory Appl.* **45** (1985) 41–51.

[5] D. T. Connolly, An improved annealing scheme for the QAP, *Eur. J. Oper. Res.* **46** (1990) 93–100.

[6] L. Constantinou, K. Bagherpour, R. Gani, J. A. Klein, D. T. Wu, Computer aided product design: problem formulations, methodology and applications, *Comput. Chem. Engin.* **20** (1996) 685–702.

[7] L. Constantinou, R. Gani, New group contribution method for estimating properties of pure compounds, *AIChE J.* **40** (1994) 1697–1710.

[8] D. Delamarre, B. Virot, Simulated annealing algorithm: technical improvements, *RAIRO Oper. Res.* **32** (1998) 43–73.

[9] K. A. Dowsland, Simulated annealing, in: C. R. Reeves (Ed.) *Modern Heuristic Techniques for Combinatorial Problems*, Wiley, New York, 1993, pp. 20–69.

[10] R. Gani, A. Fredenslund, Computer aided molecular and mixture design with specified property constraints, *Fluid Phase Equil.* **82** (1993) 39–46.

[11] J. Gasteiger, C. Jochum, An algorithm for the perception of synthetically important rings, *J. Chem. Inf. Comput. Sci.* **19** (1979) 43–48.

[12] C. Graffigne, Parallel annealing by periodically interacting multiple searches: an experimental study, in: R. Azencott (Ed.), *Simulated Annealing: Parallelization Techniques*, Wiley, New York, 1992, pp. 47–79.

[13] B. Hajek, Cooling schedules for optimal annealing, *Math. Oper. R.* **13** (1988) 311–329.

[14] J. Heintz, J. P. Belaud, N. Pandya, M. T. Dos Santos, V. Gerbaud, Computer aided product design tool for sustainable product development, *Comput. Chem. Engin.* **71** (2014) 362–376.

[15] J. D. Horton, A polynomial-time algorithm to find the shortest cycle basis of a graph, *SIAM J. Comput.* **16** (1987) 358–366.

[16] K. G. Joback, R. C. Reid, Estimation of pure-component properties from group-contributions, *Chem. Eng. Commun.* **57** (1987) 233–243.

[17] S. Kirkpatrick, C. D. Gerlatt Jr, M. P. Vecchi, Optimization by simulated annealing, *Science* **220** (1983) 671–680.

[18] J. F. Krzyzaniak, P. B. Myrdal, P. Simamore, S. H. Yalkowsky, Boiling point and melting point prediction for aliphatic, non-hydrogen-bonding compounds, *Ind. Eng. Chem. Res.* **34** (1995) 2530–2535.

[19] V. Kvasnička, J. Pospíchal, Simulated annealing construction of molecular graphs with required properties, *J. Chem. Inf. Comput. Sci.* **36** (1996) 516–526.

[20] Y. Kwon, S. Kang, Y. S. Choi, I. Kim, Evolutionary design of molecules based on deep learning and a genetic algorithm, *Sci. Rep.* **11** (2021) 17304–17314.

[21] Y. Kwon , J. Lee, MolFinder: An evolutionary algorithm for the global optimization of molecular properties and the extensive exploration of chemical space using SMILES, *J. Cheminf.* **13** (2021) 24–38.

[22] P. J. M. Van Laarhoven, E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, Kluwer, Dordrecht, 1988.

[23] E. M. W. Lameijer, *Interactive Evolutionary Algorithms and Data Mining for Drug Design*, Thesis PhD Computer Science, LIACS, Leiden University, 2010.

[24] H. T. Lau, *Algorithms on Graphs*, TAB Books, Blue Ridge, 1989.

[25] C. J. Lee, Y. M. Kang, K. H. Cho, K. T. No, A robust method for searching the smallest set of smallest rings with a path-included distance matrix, *Biophys. Comput. Biol.* **106** (2009) 17355–17358.

[26] K. I. Al-Malah, Prediction of normal boiling points of hydrocarbons using simple molecular properties, *J. Adv. Chem. Engin.* **3** (2013) 1–9.

[27] J. F. Díaz Martín, J. M. Riaño Sierra, A comparison of cooling schedules for simulated annealing, in: J. R. R. Dopico, J. Dorado, A. Pazos (Eds.), *Encyclopedia of Artificial Intelligence*, IGI Global, New York, 2009, pp. 344–352.

[28] N. Metropolis, A. W. Rosenbluth, A. H. Teller, E. Teller, Equation of state calculations by fast computing machines, *J. Chem. Phys.* **21** (1953) 1087–1092.

[29] P. Moscato, J. F. Fontanari, Stochastic versus deterministic update in simulated annealing, *Phys. Lett. A* **146** (1990) 204–208.

[30] M. W. Park, Y. D. Kim, A systematic procedure for setting parameters in simulated annealing algorithm, *Comput. Ops. Res.* **25** (1998) 207–217.

[31] A. D. Rakić, J. M. Elazar, A. B. Djurišić, Acceptance-probability-controlled simulated annealing: a method for modeling the optical constants of solids, *Phys. Rev. E* **52** (1995) 6862–6867.

[32] D. Vanderbilt, S. G. Louie, A Monte Carlo simulated annealing approach to optimization over continuous variables, *J. Comput. Phys.* **56** (1984) 259–271.

[33] V. Venkatasubramanian, K. Chan, J.M. Caruthers, Evolutionary design of molecules with desired properties using the genetic algorithm, *J. Chem. Inf. Comput. Sci.* **35** (1995) 188–195.

[34] D. Weininger, SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules, *J. Chem. Inf. Comput. Sci.* **28** (1988) 31–36.

[35] D. Weininger, A. Weininger, J. L. Weininger, SMILES. 2. Algorithm for generation of unique SMILES notation, *J. Chem. Inf. Comput. Sci.* **29** (1989) 97–101.

[36] D. Weininger, SMILES. 3. DEPICT. Graphical depiction of chemical structures, *J. Chem. Inf. Comput. Sci.* **30** (1990) 237–243.

[37] A. Zamora, An algorithm for finding the smallest set of smallest rings, *J. Chem. Inf. Comput. Sci.* **16** (1976) 40–43.