Genetic Algorithm Cluster Searching for Uranium-Containing Clusters

Paul Adamson

January 23, 2015

Contents

1 Notation

1.1 Code Chunks

Three types of code chunks that make up the software package are presented in this specification and delineated with the appropriate keyword in italics: *Source, Helper, or Test*. The filename containing the code chunk is given in parenthesis following the keyword. A brief description of the code chunk is also listed. The actual code is represented with a fixed-width font where keywords are bold and comments are italicized. An example helper code chunk is listed below.

Helper (testFunctions.chpl). Provide the functions used in the tests.

```
proc f1(x:real):real {
  return x**3;
}
proc f2(x:real):real {
  return 1/x;
}
proc f3(x:real):real {
  return x;
}
```

1.2 Examples

Examples of how to use the software are also provided and delineated with the keyword *Example* followed by a description of the use case. Here is an example of an *Example*:

Example. The following line of code calls the function `leftRectangleIntegration` to perform the integral of the function `f1` over the interval $[1.0, 4.0]$ using the left rectangle method with 100 subdivisions and stores the result in the variable `result`.

```
var result: real = leftRectangleIntegration(a = 1.0, b = 4.0, N = 100, f = f1);
```

1.3 Text Boxes

Different color text boxes are used to highlight *TODO's, Notes, Rationales, Open Issues,* and *Futures* throughout the specification. Examples of these text boxes along with definitions for each of these terms is given below:

> *TODO*. Things that need to be done for this version of the software.

Note. Something of note that does not fit into any other category.

Rationale. An explanation for a particular design choice.

Open issue. Issue that we do not know how to handle.

Future. Issue or feature that we have a story about, but which is not yet fully-designed or implemented.

2 Organization

This specification is organized as follows:

Chapter 1 Notation, introduces the notation that is used throughout the specification.

Chapter 2 Organization, describes the contents of each of the chapters within this specification.

Chapter 3 Requirements, scope and functional requirements for the genetic algorithm cluster search code that we will develop.

3 Requirements

3.1 Scope

The scope of this application is the search for optimized geometries of clusters and nanoparticles using genetic algorithms to drive quantum chemistry calculations. The basic approach follows Johnston.[1]

3.2 Functional Requirements

The code shall do the following:

R1 generate an initial generation of clusters by choosing the x, y, and z coordinates randomly and relaxing them to local minima using the quasi-Newton L-BFGS minimisation routine

for a generation, record the lowest and highest values of the potential energy; compute the fitness using dynamic fitness scaling with

- **R2.1** an exponential fitness function
- **R2.2** a linear fitness function
- **R2.3** a hyperbolic tangent fitness function

from a generation with assigned fitness values, select parents for crossover using

- **R3.1** roulette wheel selection
- **R3.2** tournament selection

R4 carry out crossover for a given set of parents, generating a predetermined number of offspring, using a variant of the cut and splice crossover operator of Deaven and Ho followed by relaxation to a local minimum using the quasi-Newton L-BFGS routing

for a set of offspring, apply a mutation with a predetermined probability, and relax any "mutant" clusters using the L-BFGS minimisation routine; the applied mutation scheme is

- **R5.1** atom displacement
- **R5.2** twisting
- **R5.3** cluster replacement
- **R5.4** atom permutation

R6 maintain diversity in the offspring population by removing the higher energy offspring of any pair with difference in energy of less than some predetermined value

R7 select a next generation of predetermined size from the lowest energy (highest fitness) clusters selected from the set containing the previous generation, the new offspring, and the new mutants

R8 repeat crossover, mutation, and selection for a specified number of generations or until convergence (range of cluster energies in the population has not changed for a prescribed number of generations)

A Requirements Traceability Matrix

Table A.1: Requirement traceability matrix.

Requirement	Specification
R1	
R2.1	
R2.2	
R2.3	
R3.1	
R3.2	
R4	
R5.1	
R5.2	
R5.3	
R5.4	
R6	
R7	
R8	

Index

Bibliography

[1] Roy L. Johnston. Evolving better nanoparticles: Genetic algorithms for optimising cluster geometries. *Dalton Trans.*, pages 4193–4207, 2003.