Contents

1 Introduction .......................... 1
  1.1 Capabilities ........................................... 1
  1.2 The Fine Print ...................................... 1

2 Installation ........................... 3
  2.1 Prerequisites ................................. 3
  2.2 Compilation and installation ................. 3

3 Invoking niedoida ..................... 5
  3.1 Command-line interface ...................... 5
  3.2 Configuration file ............................ 5
    3.2.1 Configuration settings ..................... 5
  3.3 Input description ..................... 6
    3.3.1 Comments ........................................ 6
    3.3.2 Parameter definitions ..................... 6
  3.4 Parameters ................................. 8
  3.5 Examples ................................. 10

A License ................................ 13

B HDF5 License ............................. 15

C Boost License ............................ 17

D Pearls of Wisdom ...................... 19
List of Tables

3.1 Input parameters. ....................................... 8
3.2 SCF parameters. ........................................ 9
3.3 Units. .................................................. 9
List of Figures
List of examples

3.1 A comment .................................................. 6
3.2 Simple parameters ........................................ 7
3.3 A compound parameter ...................................... 7
3.4 A list parameter ............................................. 8
3.5 Water molecule .............................................. 11
3.6 Ammonia molecule .......................................... 11
Chapter 1

Introduction

Niedoida[1] is a general-purpose quantum-chemical package, at the moment in the pupal state. For up-to-date information about niedoida check http://www.chemia.uj.edu.pl/~niedoida/en/.

1.1 Capabilities

Currently niedoida allows for SCF calculations at the Hartree-Fock level of theory. The implemented schemes are

- RHF,
- UFH,
- ROHF (in Longuet-Higgins-Pople approximation[2, 3]).

For all these methods, following properties are available:

- energy,
- molecular obitals,
- Mulliken[4, 5, 6, 7] and Löwdin population analyses,

1.2 The Fine Print

Niedoida and associated documentation is distributed according to the license allowing you to use it only for conducting scientific research, and without any warranty. For details see Appendix A.

For each publication using results obtained by running niedoida, we request citation including the name of the program, it’s version and all authors. For the current version the required citation is:

Niedoida 0.1, Grzegorz Mazur, Marcin Makowski, Witold Piskorz, Łukasz Cwiklik, Mariusz Sterzel, Mariusz Radon, 2004
Additionally, binary distribution of niedoidais using, among others, HDF5 and Boost libraries. Their licenses are reproduced in Appendices C and ??, respectively.
Chapter 2

Installation

This chapter covers all the information necessary to compile and install niedoida.

2.1 Prerequisites

Before you start installing niedoida, you need a few additional software packages necessary to compile it. The full list of dependencies is

- **Tools**
  - python version 2.4 ([http://www.python.org/](http://www.python.org/))
  - scons version 0.96 ([http://www.scons.org/](http://www.scons.org/))
  - TE\TeX\ system, eg. te\TeX\ ([http://www.tug.org/te\TeX/](http://www.tug.org/te\TeX/))

- **Libraries**
  - boost version 1.32.0 ([http://www.boost.org/](http://www.boost.org/))
  - HDF5 version 1.6.3.patch ([http://hdf.ncsa.uiuc.edu/HDF5/](http://hdf.ncsa.uiuc.edu/HDF5/))

- **Optional libraries**

2.2 Compilation and installation

To compile niedoida, run `scons` in the main directory of the program. If needed, `scons` may be parametrized with compilation options.

It is possible to set several compilation options. They may specify compilation mode and list of directories being searched by the compiler to fulfill external dependencies. Full list of options, and their default values can be obtained by running `scons -h` in the main directory of the program.
CHAPTER 2. INSTALLATION

Running `scons` generates all libraries and programs comprising `niedoida`, and full documentation in the PDF format. Additionally, basic tests of most of the components are performed during compilation.

Running `scons install` installs the program, libraries and header files.

More information about scons can be found at http://www.scons.org/.
Chapter 3

Invoking niedoida

The chapter covers all the information necessary to prepare input data and run niedoida. It is assumed that the program is already installed and configured.

3.1 Command-line interface

niedoida provides typical command-line interface. It allows for running the program with input being read either from file, or from the standard input.

Running niedoida without any arguments causes the program to read input from the standard input, and write results to the standard output. If an argument is provided, it is treated as the input file name. In this case results are written to a file. The results file is created in the same directory and has the same name as the input file, but the extension is changed to .log.

3.2 Configuration file

Default configuration is stored in niedoida during compilation. When running niedoida, it reads the global configuration file (if it exists) and then local configuration file (if it exists). Setting a configuration parameter in any of them overrides the previous value.

Configuration files format is free-form. Spaces, tabs and line breaks are uniformly treated as whitespaces, except for literal strings and comments. Configuration file may contain parameter definitions and comments.

Global configuration file is named niedoida.cfg, and is stored in directory <prefix>/share/niedoida. Default path to the global configuration file is /usr/local/share/niedoida.

Local configuration file is named niedoida.cfg, and is searched for in the same directory in which the input file is located. If input is read from standard input, local configuration file is searched for in the current working directory.

3.2.1 Configuration settings

basis_set Parameters set defining where and in which format atomic basis sets descriptions are stored. Each entry has the following form
and the whole set

```
basis_set = {
  <entry_1>,
  <entry_2>,
  ...
  <entry_n>
};
```

**scratch_dir** Parameter defining in which directory temporary files are created. Default is /tmp.

### 3.3 Input description

The input format is free-form. Spaces, tabs and line breaks are uniformly treated as whitespaces, except for literal strings and comments. Input may contain parameter definitions and comments.

#### 3.3.1 Comments

Comments may appear anywhere in the input. They start with the # character, and end at the end of the line. An example comment is shown in Example 3.1.

Comments do not influence the calculations in any way, and can be used to write notes into the input file.

```
# no comments
```

Example 3.1: A comment

#### 3.3.2 Parameter definitions

Parameters control calculations performed by niedoida. By setting them to specific values you decide what and how is calculated when the program is run.

Parameters may be of simple, compound or list type. All parameter definitions have the form

```
<parameter_name>=<parameter_value>;
```

The order of parameter definitions is not significant.\(^1\)

\(^1\)Except for the inputs where the same parameter appears more than once. In such cases only the last definition is effective. This (mis)feature should not be relied upon, and it is planned that in the future versions of niedoida an attempt to define the same parameter more than once will be reported as error.
3.3. INPUT DESCRIPTION

Simple parameters

Simple parameter values are integer or real numbers, identifiers, or literal strings. Examples are shown in Example 3.2.

```plaintext
# a string parameter
title = "an example job";

# an identifier parameter
run_type = single_point;

# an integer parameter
charge = -1;

# a real parameter
energy_threshold = 1e-6;
```

Example 3.2: Simple parameters

Compound parameters

Compound parameters are collections of other parameters. Their definitions have the form

```
<compound_parameter_name> = {
    <parameter_name_1> = <parameter_value>;
    <parameter_name_2> = <parameter_value>;
    # ...
    <parameter_name_n> = <parameter_value>;
};
```

where the embedded parameters may be of any type.

An example is shown in Example 3.3.

```plaintext
scf = {
    method = rhf;
    energy_threshold = 1e-6;
    density_threshold = 1e-6;
    convergence_accelerator = diis;
};
```

Example 3.3: A compound parameter

List parameters

List parameters are collections of values. Their definitions have the form
CHAPTER 3. INVOKING NIEDOIDA

\(<\text{list\_parameter\_name}> = \{ \\
\quad \text{<value\_1>}, \\
\quad \text{<value\_2>}, \\
\quad \# \ldots \\
\quad \text{<value\_n>} \\
\};

An example is shown in Example 3.4.

\begin{verbatim}
atoms = { \\
  o 0.0000000 0.24618131 0.0000000, \\
  h 1.4326629 -0.95521837 0.0000000, \\
  h -1.4326629 -0.95521837 0.0000000 \\
};
\end{verbatim}

Example 3.4: A list parameter

3.4 Parameters

For complete list of parameters see Tab. 3.1. All of them, except for \texttt{basis\_set} and \texttt{atoms}, are optional.

Skipping an optional parameter means that the default value is assigned to it. Note: default values are constant, and do not depend on other parameters values. No attempt is made to adjust skipped parameters to those specified in input. This means, for example, that setting multiplicity to 1, and not specifying SCF method different than default (RHF) is signalled as input error.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>string</td>
<td>&quot;&quot;</td>
<td>Title of the job.</td>
</tr>
<tr>
<td>run_type</td>
<td>enum</td>
<td>\texttt{single_point}</td>
<td>Type of the job. Allowed value is \texttt{single_point}.</td>
</tr>
<tr>
<td>basis_set</td>
<td>enum</td>
<td></td>
<td>Basis set.</td>
</tr>
<tr>
<td>atoms</td>
<td>list</td>
<td></td>
<td>List of the atoms comprising the system and their coordinates.</td>
</tr>
<tr>
<td>scf</td>
<td>compound</td>
<td></td>
<td>Set of parameters controlling the SCF process. See Tab. 3.2</td>
</tr>
<tr>
<td>units</td>
<td>compound</td>
<td></td>
<td>Specifies the (physical) units system to use for reading input and printing results. See Tab. 3.3</td>
</tr>
</tbody>
</table>

Table 3.1: Input parameters.
### 3.4. PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>enum</td>
<td>rhf</td>
<td>Specifies the type of SCF process. Allowed values are: rhf (restricted Hartree-Fock), rohf (restricted open-shell Hartree-Fock) and uhf (unrestricted Hartree-Fock)</td>
</tr>
<tr>
<td>max_no_iterations</td>
<td>positive integer</td>
<td>30</td>
<td>Maximal number of iterations.</td>
</tr>
<tr>
<td>energy_threshold</td>
<td>positive real</td>
<td>$10^{-5}$</td>
<td>SCF is not considered converged until the energy difference between two consecutive steps is larger than the threshold.</td>
</tr>
<tr>
<td>density_threshold</td>
<td>positive real</td>
<td>$10^{-5}$</td>
<td>SCF is not considered converged until the density difference between two consecutive steps is larger than the threshold.</td>
</tr>
<tr>
<td>convergence_accelerator</td>
<td>enum</td>
<td>diis</td>
<td>The convergence accelerator to use in SCF process. Allowed values are none and diis.</td>
</tr>
<tr>
<td>initial_guess</td>
<td>string</td>
<td></td>
<td>Allowed values are: core_hamiltonian, fragments or ??</td>
</tr>
</tbody>
</table>

Table 3.2: SCF parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>energy</td>
<td>enum</td>
<td>ev</td>
<td>Energy unit to use. Allowed values are hartree and ev.</td>
</tr>
<tr>
<td>length</td>
<td>enum</td>
<td>bohr</td>
<td>Length unit to use. Allowed values are bohr and angstrom.</td>
</tr>
</tbody>
</table>

Table 3.3: Units.
3.5 Examples

Example 3.5 shows an input for energy calculations of water molecule. Let’s analyze it closer.

```
title = "water molecule";
```

Sets the job title. It does not influence the calculations, but is reproduced in the output. Therefore, it can be used to convey information which makes it easier to identify or analyze output later.

```
run_type = single_point;
```

Causes the calculations to be performed only for geometry given in input. It is the default, so skipping the line would not change the calculations.

```
basis_set = sto-3g;
```

Sets the atomic orbitals basis to be used in calculations.

```
units = {
    length = bohr;
    energy = hartree;
};
```

Determines units used to interpret input, and units which will be used in output. In this case all lengths and energies are in atomic units.

```
atoms = {
    o 0.0000000 0.24618131 0.00000000,
    h 1.4326629 -0.95521837 0.00000000,
    h -1.4326629 -0.95521837 0.00000000
};
```

Determines the stoichiometry and geometry of the molecule for which calculations should be performed. In this case it is a water molecule, and geometry is close to optimal.
3.5. EXAMPLES

Example 3.5: Water molecule

```plaintext
title = "water molecule";
run_type = single_point;
basis_set = sto-3g;
units = {
    length = bohr;
    energy = hartree;
};
atoms = {
    o 0.0000000 0.24618131 0.00000000,
    h 1.4326629 -0.95521837 0.00000000,
    h -1.4326629 -0.95521837 0.00000000
};
```

Example 3.6: Ammonia molecule

```plaintext
title = "ammonia molecule";
run_type = single_point;
basis_set = sto-3g;
units = {
    length = angstrom;
    energy = hartree;
};
atoms = {
    n 0.0000000000 0.0000000000 0.5841387237,
    h -0.4702866428 0.8145603594 0.1580719249,
    h -0.4702866428 -0.8145603594 0.1580719249,
    h 0.9405732855 0.0000000000 0.1580719249
};
```
Appendix A

License

Copyright (c) 2004, 2005 Grzegorz Mazur, Marcin Makowski, Witold Piskorz, Łukasz Cwiklik, Mariusz Sterzel, Mariusz Radoń

LICENSE

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to use it for conducting scientific research.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
Appendix B

HDF5 License

Copyright Notice and Statement for NCSA Hierarchical Data Format (HDF) Software Library and Utilities

NCSA HDF5 (Hierarchical Data Format 5) Software Library and Utilities
Copyright 1998, 1999, 2000, 2001, 2002, 2003, 2004 by the Board of Trustees of the University of Illinois. All rights reserved.

Contributors: National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign (UIUC), Lawrence Livermore National Laboratory (LLNL), Sandia National Laboratories (SNL), Los Alamos National Laboratory (LANL), Jean-loup Gailly and Mark Adler (gzip library).

Redistribution and use in source and binary forms, with or without modification, are permitted for any purpose (including commercial purposes) provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or materials provided with the distribution.

3. In addition, redistributions of modified forms of the source or binary code must carry prominent notices stating that the original code was changed and the date of the change.

4. All publications or advertising materials mentioning features or use of this software are asked, but not required, to acknowledge that it was developed by the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign and to credit the contributors.

5. Neither the name of the University nor the names of the Contributors may be used to endorse or promote products derived from this software without specific prior written permission from the University or the Contributors,
as appropriate for the name(s) to be used.

6. THIS SOFTWARE IS PROVIDED BY THE UNIVERSITY AND THE CONTRIBUTORS "AS IS"
WITH NO WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED. In no event
shall the University or the Contributors be liable for any damages
suffered by the users arising out of the use of this software, even if
advised of the possibility of such damage.

--------------------------------------------------------------------------
Portions of HDF5 were developed with support from the University of
California, Lawrence Livermore National Laboratory (UC LLNL).
The following statement applies to those portions of the product
and must be retained in any redistribution of source code, binaries,
documentation, and/or accompanying materials:

This work was partially produced at the University of California,
Lawrence Livermore National Laboratory (UC LLNL) under contract no.
W-7405-ENG-48 (Contract 48) between the U.S. Department of Energy
(DOE) and The Regents of the University of California (University)
for the operation of UC LLNL.

DISCLAIMER:
This work was prepared as an account of work sponsored by an agency
of the United States Government. Neither the United States
Government nor the University of California nor any of their
employees, makes any warranty, express or implied, or assumes any
liability or responsibility for the accuracy, completeness, or
usefulness of any information, apparatus, product, or process
disclosed, or represents that its use would not infringe privately-
owned rights. Reference herein to any specific commercial products,
process, or service by trade name, trademark, manufacturer, or
otherwise, does not necessarily constitute or imply its endorsement,
recommendation, or favoring by the United States Government or the
University of California. The views and opinions of authors
expressed herein do not necessarily state or reflect those of the
United States Government or the University of California, and shall
not be used for advertising or product endorsement purposes.

--------------------------------------------------------------------------
Appendix C

Boost License

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
Appendix D

Pearls of Wisdom

C++: an octopus made by nailing extra legs onto a dog.  
-- Anonymous

FORTRAN was the language of choice for the same reason that three-legged races are popular.  
-- Ken Thompson

There's no Moore law for the productivity of programmers.  
-- Robert Dewar

* Me: My brain is burnt out after trying to follow what the iterators have to do.  
* Joerg: That's Dr. Stepanov's gift for us ;-)  
  -- Boost developer

DFT is assuming a local external potential ....  
-- Bernd Schimmelpfennig

Failure is not an option. It comes bundled with your Microsoft product.  
-- Derek Harkness

A bug in the code is worth two in the documentation.  
-- Anonymous

If the only thing you know is a hammer, everything looks like a nail.  
-- Anonymous
Bibliography


Index

boost, 3
gcc, 3
HDF5, 3
MPI
   LAM/MPI, 3
   MPICH2, 3
population analysis, 1
   Löwdin, 1
   Mulliken, 1
python, 3
RHF, 1
ROHF, 1
rubber, 3
scons, 3
TeX, 3
UHF, 1
valence analysis, 1
   Gopinathan-Jug, 1
   Mayer, 1
   Nalewajski-Mrozek, 1