ARL-TR-7304 • JUN 2015

US Army Research Laboratory

Energetic Materials Optimization via Constrained Search

by Berend Christopher Rinderspacher

NOTICES

Disclaimers

ARL-TR-7304 • JUN 2015

US Army Research Laboratory

Energetic Materials Optimization via Constrained Search

by Berend Christopher Rinderspacher
Weapons and Materials Research Directorate, ARL

REPORT DOCUMENTATION PAGE	***Form Approved OMB No. 0704-0188***

1. REPORT DATE *(DD-MM-YYYY)* June 2015	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)** 2013-2014

4. TITLE AND SUBTITLE Energetic Materials Optimization via Constrained Search	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Berend Christopher Rinderspacher	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) US Army Research Laboratory ATTN: RDRL-WMM-G Aberdeen Proving Ground, MD 21005	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-7304
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR’S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

primary author’s email: <berend.c.rinderspacher.civ@mail.mil>.

14. ABSTRACT

The design and optimization of energetic materials is a highly complex problem subject to several constraints. For performant propellants, the target of maximization is the Chapman-Jouguet (CJ) pressure. Yet, the material has to remain synthetically viable and retain the detonation characteristics that make it an explosive, such as the shock velocity and the total detonation energy.

We apply a primal-dual interior-point algorithm of first order on a hypertorus with multiple Lagrange multipliers. The energetic characteristics are computed using density functional theory for input into heuristic programs. The method is highly efficient and produces viable candidates. We compare the results to a maximization of total detonation energy under synthetic viability constraint.

15. SUBJECT TERMS

chemical compound space, discrete optimization, energetic materials, synthetic viability, multiply constrained optimization

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** B Christopher Rinderspacher
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	34	**19b. TELEPHONE NUMBER (Include area code)** 410-306-2811

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

1,3,5-trinitroperhydro-1,3,5-triazine (RDX) and octahydro-1,3,5,7-tetranitro-1,3,5,7-tetrazocine (HMX) are standard military explosives. RDX and HMX have detonation velocities of more than 8.8 km/s, Chapman-Jouguet (CJ) pressures above 33 GPa, and total energies of detonation above 10 kJ/mL. The performance of a given propellant is largely dominated by its CJ pressure. It is therefore desirable to find explosives with comparable detonation characteristics, i.e., shock velocity and total energy of detonation, to RDX or HMX while exhibiting increased ability to propel loads. This enables less propellant or higher final velocities in applications, but the usefulness is curtailed by the synthetic accessibility, which may result in higher costs.

Inverse molecular design (IMD) is a method for materials optimization well suited for the exploration of energetic materials. Of late, IMD beyond the purview of drug design has enjoyed increasing popularity. Examples can be found in protein design[1–4] or high-hyperpolarizability materials.[5–7] The design problem is complicated by the vastness of possible chemicals, termed chemical space. This space can be viewed as combinatorially complex, e.g., $20^8 \approx 2.6 \cdot 10^{10}$ octa-peptides of the naturally occurring amino acids exist alone. As a consequence, a variety of methods have been developed for the discrete optimization in chemical subspaces.[8–15] The continuous optimization of chemicals used in the linear combination of atomic potentials (LCAP) method[16,17] and the variation-of-particles density-functional-theoretical (VP-DFT) method introduces an important concept for dealing with the inherent roughness of chemical compound space.[18–20] LCAP and VP-DFT interpolate continuously between the Hamiltonians of various chemical species. Furthermore, recently an investigation into the reasons why chemical optimization is, relatively speaking, “easy” utilized probability distributions and expectations on the control variables to arrive at its conclusion.[21,22] Recently, it was discovered that indeed all properties can be optimized efficiently only using the charge density with as-yet unknown functionals.[23] All previous optimizations were executed using at most single constraints. In this contribution, we introduce a flexible method for multiple nonlinear (inequality) constraints applied to a combinatorially complex search space of energetic materials.

2. Computational Chemistry Protocol

All quantum-mechanical computations were performed using *Gaussian 09*.[24] All geometries were preoptimized with B3LYP/3-21G under constraints as described elsewhere[25] followed by a geometry optimization using B3LYP/3-21G* (see Listing 1 in the Appendix for specifics).

Before further analysis, the conformational space was explored to ensure low-energy species.[25] Natural-bond-order analysis was used to determine connectivity and create SMILES (simplified molecular-input line-entry system identifiers) for input into SYLVIA for synthetic viability analysis.[26] The heats of formation and density were computed following Byrd.[27] These served as input to Cheetah[28] for energetic characterization. Listing 2 summarizes these steps.

3. Optimization Methodology

Our optimization problem is formulated as a constrained maximization:

$$\begin{aligned} &\max_{x \in CCS} P(x) \\ s.t.: &\ TED(x) - 9.75 \geq 0 \\ &\ SV(x) - 9 \geq 0 \\ &\ 5 - SA(x) \geq 0, \end{aligned} \tag{1}$$

where $TED(x)$ is the total energy of detonation (TED) of compound x from the chosen chemical subspace (CCS) of chemical compound space discussed later, $SV(x)$ is the shock velocity, $P(x)$ is the CJ pressure, and $SA(x)$ is the synthetic accessibility. We reformulate problem 1 via nonnegative Lagrange multipliers $\lambda \in \mathbb{R}^3_+$ for the 3 constraints to the augmented Lagrangian function $\mathcal{L}(x, \lambda) := P(x) - \lambda C(x)$ as a constrained min-max problem,

$$\max_{x \in CCS} \min_{\lambda \in \mathbb{R}^3_+} P(x) - \lambda^T C(x), \tag{2}$$

where $C(x)$ is the vector of constraint violations, i.e., $\eta(9.75 - TED(x)), \eta(9 - SV(x)), \eta(SA(x) - 5)$, and η is the violation function:

$$\eta(y) = \begin{cases} y, & y \geq 0 \\ 0, & \text{otherwise} \end{cases}.$$

We solve problem 2 alternatingly between the primal variable(s) x and the dual viariables λ. For fixed λ, x is varied along each substitution direction until no improvement is found. After all search directions have been searched, the minimization with respect to λ is performed using the dual function $P_{(d)}(\lambda) = \max_{x \in CCS} \mathcal{L}(x, \lambda)$ and its approximation

$$P'_{(d)}(\lambda) = \max_{x \in CCS'} \mathcal{L}(x, \lambda), \tag{3}$$

where CCS' is the subset of visited, i.e., already computed, molecules. Due to the discrete nature of CCS', $P'_{(d)}$ is a piece-wise linear function of λ with general derivative $-C(x(\lambda))$, where $x(\lambda) := \arg\max_{y \in CCS'} \mathcal{L}(y, \lambda)$. Due to the piecewise linearity, λ only changes meaningfully when $x(\lambda) \neq x(\lambda')$. Thus, whenever $C(x) \neq 0$, λ is updated cumulatively by:

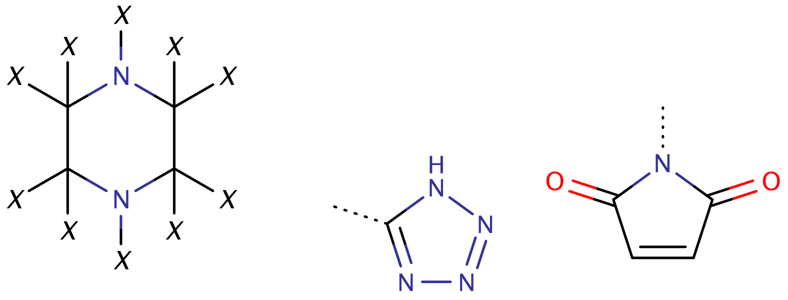
$$\Delta\lambda = \alpha C(x^*) \max\{\underline{\lambda}, 0\} \tag{4}$$

$$\underline{\lambda} = \inf\{\lambda' \in \mathbb{R}_+ | \mathcal{L}(x, \lambda^* + \lambda' C(x^*)) > \mathcal{L}(x^*, \lambda^* + \lambda' C(x^*))\} \tag{5}$$

where $\alpha > 1$, x^* is the currently active molecule, and λ^* are the currently active Lagrange multipliers.

Choosing the update of λ along the constraint violation direction $C(x^*)$ results in a steepest descent update with a conservative step size. The step size is chosen such that either x^* violates constraints the least as well as maximizing P or that there exists $x' \in CCS'$ with an improved $\mathcal{L}(x', \lambda + \Delta\lambda) > \mathcal{L}(x^*, \lambda + \Delta\lambda)$.

The chemical subspace that we investigate is depicted in Fig. 1. Each X on **1** represents a substitution site for which one of 10 chemical groups is attached. The overall search space thus encompasses 10^{10} molecules. Each such substitution site represents an independent search direction.[14] The substitutions are enumerated and thus can be viewed as the integer positions on a circle. The optimization proceeds with a local line search for each substitution site in the prescribed order of sites. When neighboring substitutions in the current direction to the current iterate are inferior, the line search is halted and the next direction is searched.



1 **2** **3**

Fig. 1 Optimization framework. Each X may be replaced by -H, -C(O)OMe, -NO$_2$, -C(O)Me, -C(NO$_2$)$_2$H, -C(NO$_2$)$_3$, -CH$_2$OH, -CN, 2, or 3 for a total of 10^{10} possible molecules.

Since generally there is no a priori knowledge of the proper search order of each substituent, the initial assignment is generally unsuited for smooth optimization. To mitigate this problem, the enumeration of substituents is reassessed after each full cycle of local searches in each direction. For each substituent the average arc tangent of the Lagrangian value is computed,

$$\overline{x}_i^{(j)} = \frac{1}{\#\{x \in CCS' | x_i = j\}} \sum_{x \in CCS' | x_i = j} \arctan \mathcal{L}(x, \lambda), \tag{6}$$

which is subsequently used to order the substituents around the circle starting with the lowest scoring substituent and placing substituents alternatingly to the left and right in ascending order. The resulting order produces a smooth ordering that is monotonically increasing until the maximum is reached, and then monotonically decreasing until the minimum is reached, or vice versa depending on which part of the circle one starts and in which direction one goes. The arc tangent is used because some computations may fail due to convergence issues in geometries or general instability of the molecule. In such a case, a molecule is mapped to a Lagrangian value of $-\infty$. The general program flow is visualized in Fig. 2.

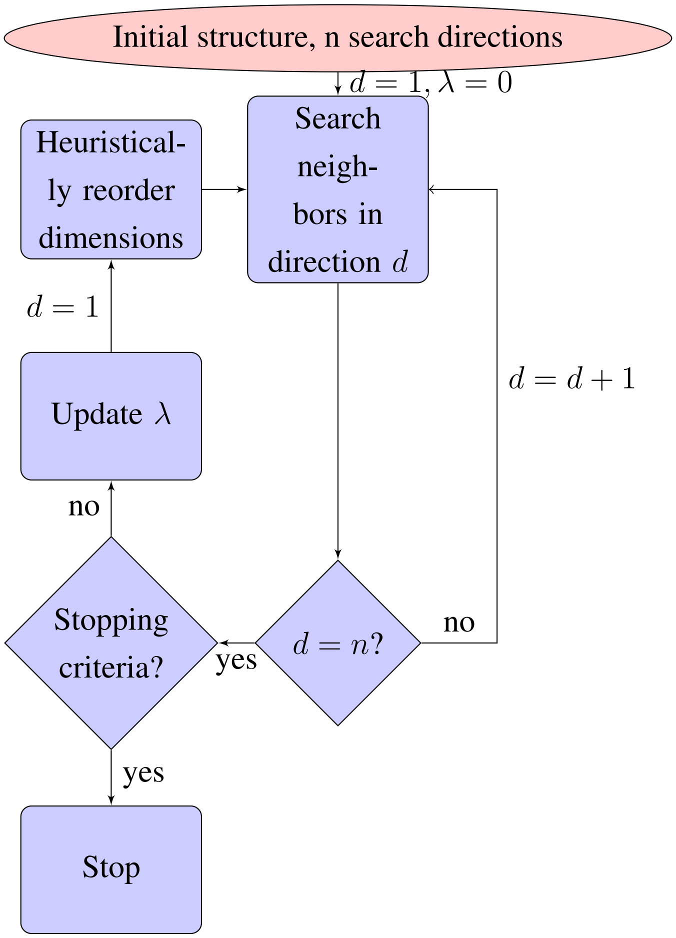
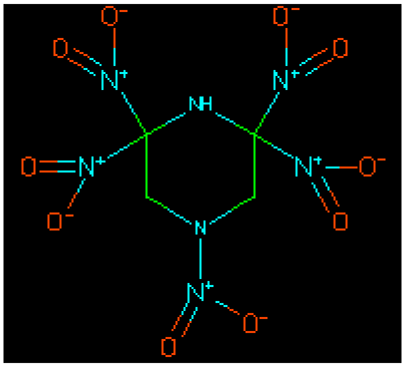


Fig. 2 **Flowchart of algorithm**

4. Results and Discussion

If TED is optimized and only SA is constrained, then nitration of all possible positions occurs. Steric hindrances select these positions when the maximum number of nitro-groups has been reached.

A different picture emerges under full constraints optimizing for CJ pressure. In this case, the optimization still primarily nitrates, but CJ pressure is attenuated by minimizing steric interference leading to exceptionally high CJ-pressure for moderate synthetic effort. The final compound **4** (see Fig. 3) exhibits a CJ pressure of 40.791 GPa, a shock velocity of 9.436 km/s, a total energy of detonation of 12.447 kJ/mL, and a synthetic accessibility score of 4.570 out of 10.



4

Fig. 3 Best candidate found

5. Conclusions

We have developed an efficient algorithm for handling multiple constraints in the optimization of substitutional subspaces of chemical compound space. We propose novel energetic materials (see **4** in Fig. 3). Analysis of the trajectory suggests that molecules with the general feature of sterically relaxed nitro-groups on the ring are particularly promising for experimental evaluation. While increased TED and shock velocity directly correlate with the number of nitro-groups, the CJ pressure decreases with internal steric stress. All highly performing molecules include a substitution on one of the ring nitrogens while nitrating the far ring carbons.

6. References

1. Bolon DN, Mayo SL. Enzyme-like proteins by computational design. Proc Nat Acad Sci USA. 2001;98(25):14274-14279.

2. Havranek JJ, Harbury PB. Automated design of specificity in molecular recognition. Nat Struct Biol. 2003;10(1):45-52.

3. Park S, Xi Y, Saven JG. Advances in computational protein design. Curr Opin Struct Biol. 2004;14(4):487-494.

4. Mang NG, Zeng C. Reference energy extremal optimization: A stochastic search algorithm applied to computational protein design. J Comp Chem. 2008;29(11):1762-1771.

5. Dudiy SV, Zunger A. Searching for alloy configurations with target physical properties: Impurity design via a genetic algorithm inverse band structure approach. Phys Rev Lett. 2006;97(4):046401.

6. Franceschetti A, Dudiy SV, Barabash SV, Zunger A, Xu J, van Schilfgaarde M. First-principles combinatorial design of transition temperatures in multicomponent systems: the case of Mn in GaAs. Phys Rev Lett. 2006;97(4):047202.

7. Piquini P, Graf PA, Zunger A. Band-gap design of quaternary (In,Ga)(As,Sb) semiconductors via the inverse-band-structure approach. Phys Rev Lett. 2008;100(18):186403.

8. Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. Science. 1983;220(4598):671-680.

9. Bohachevsky IO, Johnson ME, Stein ML. Generalized simulated annealing for function optimization. Technometrics. 1986;28(3):209-217.

10. Muhlenbein H, Gorgesschleuter M, Kramer O. Evolution algorithms in combinatorial optimization. Parallel Comput. 1988;7(1):65-85.

11. Desmet J, Demaeyer M, Hazes B, Lasters I. The dead-end elimination theorem and its use in protein side-chain positioning. Nature. 1992;356(6369):539-542.

12. Allen BD, Mayo SL. Dramatic performance enhancements for the faster optimization algorithm. J Comp Chem. 2006;27(10):1071-1075.

13. Keinan S, Paquette WD, Skoko JJ, Beratan DN, Yang WT, Shinde S, Johnston PA, Lazo JS, Wipf P. Computational design, synthesis and biological evaluation of para-quinone-based inhibitors for redox regulation of the dual-specificity phosphatase Cdc25B. Organic & Biomolecular Chemistry. 2008;6(18):3256-3263.

14. Rinderspacher BC, Andzelm J, Rawlett A, Dougherty J, Beratan DN, Yang W. Discrete optimization of electronic hyperpolarizabilities in a chemical subspace. Journal of Chemical Theory and Computation. 2009;5(12):3321-3329.

15. Weymuth T, Reiher M. Gradient-driven molecule construction: an inverse approach applied to the design of small-molecule fixating catalysts. International Journal of Quantum Chemistry. 2014;114(13):838–850.

16. Wang ML, Hu XQ, Beratan DN, Yang WT. Designing molecules by optimizing potentials. J Am Chem Soc. 2006;128(10):3228-3232.

17. Keinan S, Hu XQ, Beratan DN, Yang WT. Designing molecules with optimal properties using the linear combination of atomic potentials approach in an AM1 semiempirical framework. J Phys Chem A. 2007;111(1):176-181.

18. von Lilienfeld OA, Lins RD, Rothlisberger U. Variational particle number approach for rational compound design. Phys Rev Lett. 2005;95(15):153002.

19. von Lilienfeld OA, Tavernelli I, Rothlisberger U, Sebastiani D. Variational optimization of effective atom centered potentials for molecular properties. J Chem Phys. 2005;122(1).

20. von Lilienfeld OA, Tuckerman ME. Molecular grand-canonical ensemble density functional theory and exploration of chemical space. J Chem Phys. 2006;125(15).

21. Moore KW, Pechen A, Feng XJ, Dominy J, Beltrani VJ, Rabitz H. Why is chemical synthesis and property optimization easier than expected? Phys Chem Chem Phys. 2011;13:10048-10070.

22. Green DF. A statistical framework for hierarchical methods in molecular simulation and design. J Chem Theory Comput. 2010;6(5):1682-1697.

23. Rinderspacher BC. Chemical compound design using nuclear charge distributions. Aberdeen Proving Ground (MD): Army Research Laboratory (US); 2012. Report No.: ARL-TR-5935.

24. Gaussian 09. Wallingford (CT): Gaussian Inc. 2004 [accessed 2014 Dec 23]. http://www.gaussian.com.

25. Rinderspacher BC. Discrete optimization in chemical space reference manual. Aberdeen Proving Ground (MD): Army Research Laboratory (US); 2012. Report No.: ARL-TR-6202.

26. Boda K, Seidel T, Gasteiger J. Structure and reaction based evaluation of synthetic accessibility. Journal of Computer-Aided Molecular Design. 2007;21(6):311-325.

27. Byrd EFC. Simple computation of the heat of formation and density from theoretically predicted values. Aberdeen Proving Ground (MD): Army Research Laboratory (US); 2012. Report No.: ARL-TR-6206.

28. Fried L, Souers P. CHEETAH: a next generation thermochemical code. Livermore (CA): Lawrence Livermore National Laboratory; 1994. Report No.: UCRL-ID-117240.

INTENTIONALLY LEFT BLANK.

Appendix. Listings

Listing 1 energy_run.sh runs the constrained geometry preoptimization and the follow-up full optimization. It returns the final coordinates.

```
#!/bin/bash
# Execution script
# remove extra files
EXEC=$PWD/g09_run
filename=`basename ${1} .dat`
mkdir -p $1

if [ ! -e "$1/energy" -a ! -e "$1/failed" ]; then
    cd $1
    echo %chk=opt.chk > pre.com
    cat ../header.com >> pre.com
    cat ../$1.zmat >> pre.com
    echo %chk=opt.chk > opt.com
    cat ../footer.com >> opt.com
    if [ ! -e "pre.log" ]; then
       NORMALEXEC=`tail pre.log | grep -o termination `
       if [ ! -z "$NORMALEXEC" -o ! -e "opt.chk" ]; then
          $EXEC pre.com pre.log
       fi
    fi
    $EXEC opt.com opt.g09_out
    NORMALEXEC=`tail opt.g09_out | grep -o Normal| awk '{print\
      $1}'`
    if [ "$NORMALEXEC" == "Normal" ]; then
        fgrep opt.g09_out -e 'SCF Done' | tail -1 | awk '{print\
          $5}' > energy
        awk -f ../logcart.awk < opt.g09_out > xyz
        python ~/bin/retrieve_zmat_from_xyz.py ../$1.zmat xyz \
            | grep c > opt.rconsts
        python ~/bin/retrieve_zmat_from_xyz.py ../$1.zmat xyz \
            | grep d > opt.rvars
        ../retrieve_zmat_g03 opt.g09_out opt
        bzip2 opt.g09_out
        bzip2 *.chk
```

```
        cd ..
        ln -s $1/energy $1.energy
        ln -s $1/opt.rvars $1.rvars
        ln -s $1/opt.rconsts $1.rconsts
    else
        echo -1000000 > result
        touch failed
        exit 1
    fi
fi
```

Listing 2 proprty_script.sh computes the properties and penalties of a molecule.

```
#!/bin/bash
# Execution script

EXEC=$PWD/g09_run
filename=‘basename ${1} .dat‘
export LD_LIBRARY_PATH=$HOME/slash/lib:$LD_LIBRARY_PATH
export PATH=$PATH:$HOME/pkgs/SYLVIA/bin

mkdir -p $1

if [ ! -e "$1/failed" -a -e "$1/xyz" ]; then
    cd $1
    if [ ! -e "result" ];  then
        bunzip2 *.g09_out.bz2
        bunzip2 *out.bz2
        if [ ! -e "nbo.out" ]; then
           cp ../nbo.com nbo.com
           tail -n +3 xyz >> nbo.com
           echo >> nbo.com
           $EXEC nbo.com nbo.out
        fi
        if [ ! -e "smile.smi" ]; then
           ../NBOtoSMILE nbo.out smile.smi
           BLA=‘cat smile.smi‘
```

```
  echo $BLA";" > smile.smi
fi
if [ ! -e "sascore.smi" ]; then
   sylvia smile.smi sascore.smi
fi
if [ ! -e "penalty" ]; then
   cat sascore.smi | awk -F ';' '{print $3-5"\n0"}' \
    | sort -g | tail -1 > penalty
fi
if [ ! -e "heat_of_formation.out" ]; then
  rm -f heatstring*
fi
../smile2heatstring > heatstring.dat
cat xyz |tail -n +3 >> heatstring.dat
../master_densheat.s heatstring.dat
MASS=`obprop xyz | grep mol_weight | awk '{print $2}'`
COMPOSITION=`obprop xyz | grep ormula | awk '{print $2}'`
if [ ! -e "cheetah.out" ]; then
    cp ../cheeta.top $1.chi
    python ../build_cheetah_input.py heat_of_formation.out\
       density.out smile.smi >> $1.chi
    cat ../cheeta.bottom >> $1.chi
    source ~/pkgs/cheetah7.0/bin/cheetah.sh
    ~/pkgs/cheetah7.0/bin/cheetah $1.chi > cheetah.out
    fgrep cheetah.out -e 'Total energy of detonation'\
       | awk '{print -1*$6}' > TED
    fgrep summary.out -e 'Pressure' | awk '{print $3}'\
       | tail -1 > DetPressure
    fgrep summary.out -e 'Shock velocity' | awk '{print\
       $4}' | tail -1 >ShockVel
    cat TED | awk '{print 9.75-$1"\n0"}' | sort -g\
       | tail -1 >> penalty
    cat ShockVel | awk '{print 9.0-$1"\n0"}' | sort\
       -g | tail -1 >> penalty
    cp DetPressure result
fi
```

```
        bzip2 *.out
        cd ..
        ln -s $1/result $1.result
        ln -s $1/penalty $1.penalty
        cd $1
    fi
    bzip2 *.g09_out
    cd ..
else
    exit -1
fi
```

Listing 3 extract_results.py

```
import sys,io
def main(filename,filenamed):
        with open(filename) as f:
          line = f.readline()
          line = f.readline()
          r = line.rsplit()
        with open(filenamed) as f:
          line = f.readline()
          line = f.readline()
          r2 = line.rsplit()
        print float(r[len(r)-1])*float(r2[len(r2)-1])

if __name__ == "__main__":
    main(sys.argv[1],sys.argv[2])
```

Listing 4 build_cheeta_input.py

```
import sys
sys.path.append('/usr/people/crinders/slash/lib')
import io,openbabel,collections
NumToName = { 6:"C", 7:"N", 1:"H", 8:"O"}
def is_number(s):
    try:
        float(s)
```

```
        return True
    except ValueError:
        return False

def split_formula(formula):
        tokens = list(formula)
        last_was_number = is_number(tokens[0])
        n = 0
        w = ""
        output = ""
        for i in range(len(tokens)):
                if last_was_number:
                        if is_number(tokens[i]):
                                w += tokens[i]
                        else:
                                output+=","+w
                                w = tokens[i]
                else:
                        if is_number(tokens[i]):
                                output+=","+w
                                w=tokens[i]
                        else:
                                w+=tokens[i]
                last_was_number = is_number(tokens[i])

        output+= ","+w
        return output

def composition_of(mol):
        comp = collections.defaultdict(int)
        for obatom in openbabel.OBMolAtomIter(mol):
                comp[obatom.GetAtomicNum()] += 1
        rs = ""
        for k in comp.keys():
                rs += NumToName[k]+","+str(comp[k])+","
        if len(rs)>0:
```

```
            rs=rs[0:len(rs)-1]
        return rs

def main(filename,filenamed,composition):
        conv = openbabel.OBConversion()
        conv.SetInAndOutFormats("smi", "mol2")
        mol  = openbabel.OBMol()
        conv.AddOption("h",openbabel.OBConversion.OUTOPTIONS)
        conv.ReadFile(mol,composition)
        conv.AddOption("h",openbabel.OBConversion.OUTOPTIONS)
        mol.AddHydrogens()
        with open(filename) as f:
          line = f.readline()
          line = f.readline()
          heat = line.rsplit()
        with open(filenamed) as f:
          line = f.readline()
          line = f.readline()
          dens = line.rsplit()
        print "new␣form,␣myexplosive,␣",float(dens[len(dens)-1]),
        ",",float(heat[len(heat)-1])*4.184,",␣",composition_of(mol)

if __name__ == "__main__":
    main(sys.argv[1],sys.argv[2],sys.argv[3])
```

Listing 5 smile2heatstring

```
#!/bin/bash

# NBOtoSMILE arranges N3 to [N-]-[N+]#N
AZIDES=`obgrep2 -N '[C](-[NX2-]-[NX2+]#[NX1])' smile.smi | head -1`
# Nitro groups are always [N+](=O)[O-]
CNITROS=`obgrep2 -N '[C](-[NX3](=[OX1])[OX1-])' smile.smi | head -1`
ONITROS=`obgrep2 -N '[O](-[NX3](=[OX1])[OX1-])' smile.smi | head -1`
NNITROS=`obgrep2 -N '[N](-[NX3](=[OX1])[OX1-])' smile.smi | head -1`
# NO bound to any atom
NITROSO=`obgrep2 -N '*(-[NX2](=[OX1]))'         smile.smi | head -1`
```

```
# All SP3 carbons
CSP3=‘obgrep2 –N ’[#6^3]’                    smile.smi | head −1‘
# All hydrogens
Hs=‘obgrep2 –N ’[H]’                         smile.smi | head −1‘
# Is nitrogen but doesn’t have double  or triple bonds
# and is not negatively charged
NSP3=‘obgrep2 –N ’[N&!$([N]=∗)&!$([N]#∗)&!$([N−])]’ smile.smi |\
    head −1‘
OSP3=‘obgrep2 –N ’[OX2]’ smile.smi | head −1‘
# every carbon that is not in CSP3
CPRIME=‘obgrep2 –N ’[#6&!$([#6^3])]’ smile.smi | head −1‘
# every nitrogen that is not in NSP3
NPRIME=‘obgrep2 –N ’[n,$(N=∗),$(N#∗),NX2]’ smile.smi | head −1‘
# every oxygen that is not in OSP3
OPRIME=‘obgrep2 –N ’[O&!$([OX2])]’ smile.smi | head −1‘

echo $CSP3 $Hs $NSP3 $OSP3 $CPRIME $NPRIME $OPRIME $CNITROS\
   $NNITROS $ONITROS $AZIDES $NITROSO
```

Listing 6 Input to DiscreteOpt defining the substitution framework and search directions.

```
ChemGroup(
 (
  Z(
    (C, −3, 0.00, −2, 000.0, −1, 000.0)
    (C,  0, 1.44, −3, 000.0, −2, 000.0)
    (N,  1, 1.44,  0, 109.5, −3, 000.0)
    (C,  2, 1.44,  1, 109.5,  0, 60.0)
    (C,  3, 1.44,  2, 109.5,  1, −60.0)
    (N,  4, 1.44,  3, 109.5,  2, 60.0)
  )
  ReturnConnector()
  Connector(
   (
    (0,1,5)
    (0.7, 0.0, 0.0)
    (0.0, 109.5, 0.0)
```

```
  (0.0, 0.0, 120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (1,2,0)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, 120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (3,4,2)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, 120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (4,5,3)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, 120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
```

```
(
  (0,1,5)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, -120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (1,2,0)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, -120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (3,4,2)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, -120.0)
  (0,0,1)
  (0,0,1)
  (0,0,0)
  ()
)
(
  (4,5,3)
  (0.7, 0.0, 0.0)
  (0.0, 109.5, 0.0)
  (0.0, 0.0, -120.0)
  (0,0,1)
```

```
    (0,0,1)
    (0,0,0)
    ()
   )
   (
    (2,3,1)
    (0.7, 0.0, 0.0)
    (0.0, 109.5, 0.0)
    (0.0, 0.0, -120.0)
    (0,0,1)
    (0,0,1)
    (0,0,0)
    (240)
   )
   (
    (5,0,4)
    (0.7, 0.0, 0.0)
    (0.0, 109.5, 0.0)
    (0.0, 0.0, -120.0)
    (0,0,1)
    (0,0,1)
    (0,0,0)
    (240)
   )
  )
  allowed_groups(
   (1,2,3,4,5,6,7,8,9) # H, C(=O)OMe, NO2, Ac, C(NO2)X, CH2OH, CN,\
tetraazaCp, maleic anhydride
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
   (1,2,3,4,5,6,7,8,9)
```

```
  (1,2,3,4,5,6,7,8,9)
 )
)
(#1
 Z(
  (H, -3, 0.4, -2, 0.0, -1, 0.0)
  )
 ReturnConnector()
 Connector()
 allowed_groups()
)
 (#2
   Z(
   (C, -3, 0.84, -2, 0.0    , -1, 000.0)
   (O,  0, 1.3 , -3, 120.0  , -2, 000(180))
   (O,  0, 1.4 ,  1, 120.0  , -3, 180.0)
   (C,  2, 1.50,  0, 120.47,  1, 0(180))
   (H,  3, 1.10,  2, 109.47,  0, 180)
   (H,  3, 1.10,  2, 109.47,  4, 120)
   (H,  3, 1.10,  2, 109.47,  4,-120)
   )
   ReturnConnector()
   Connector()
   allowed_groups()
 )
 (#3
   Z(
   (N, -3, 0.74, -2, 0.0    , -1, 000.0)
   (O,  0, 1.3 , -3, 120.0  , -2, 000)
   (O,  0, 1.3 ,  1, 120.0  , -3, 180.0)
   )
   ReturnConnector()
   Connector()
   allowed_groups()
 )
 (
```

```
  Z(#4
   (C, -3, 0.70, -2, 000.0, -1, 000.00)
   (O,  0, 1.30, -3, 120.0, -2, 000.00)
   (C,  0, 1.54,  1, 120.0, -3, 180.00)
   (H,  2, 1.09,  0, 109.5,  1, 180.00)
   (H,  2, 1.09,  0, 109.5,  3, 120.00)
   (H,  2, 1.09,  0, 109.5,  4, 120.00)
  )
   ReturnConnector()
   Connector()
   allowed_groups()
)
(#5
  Z(
   (C, -3, 0.74, -2, 000.00, -1, 000)
   (N,  0, 1.10, -3, 109.47, -2, 180)
   (N,  0, 1.10, -3, 109.47,  1, 120)
   (O,  1, 1.3 ,  0, 120.0 ,  2, 160)
   (O,  1, 1.3 ,  0, 120.0 ,  3, 180.0)
   (O,  2, 1.3 ,  0, 120.0 ,  1, 160)
   (O,  2, 1.3 ,  0, 120.0 ,  5, 180.0)
   )
   ReturnConnector()
   Connector(
    (
     (0,-3,1)
     (0.7, 0.0, 0.0)
     (0.0, 109.5, 0.0)
     (0.0, 0.0, -120.0)
     (0,0,1)
     (0,0,1)
     (0,0,0)
     ()
    )
   )
   allowed_groups((1,3)) # H, NO2
```

```
)
(
   Z(#6
   (C, -3, 0.74, -2, 000.00, -1, 000)
   (H,  0, 1.10, -3, 109.47, -2, 180)
   (H,  0, 1.10, -3, 109.47,  1, 120)
   (O,  0, 1.40, -3, 109.47,  1,-120)
   (H,  3, 1.09,  0, 109.47,  1, 180)
   )
   ReturnConnector()
   Connector()
   allowed_groups()
)
(#7
 Z(
   (C, -3, 0.70, -2, 000.0, -1, 000)
   (N,  0, 1.36, -3, 170.0, -2, 180)
  )
   ReturnConnector()
   Connector()
   allowed_groups()
)
(
 Z(#8
   (C, -3, 0.70, -2, 000.00, -1, 000)
   (N,  0, 1.44, -3, 126.00, -2, 170(-180))
   (N,  1, 1.44,  0, 108.00, -3, 180)
   (N,  2, 1.44,  1, 108.00,  0, 000)
   (N,  3, 1.44,  2, 108.00,  1, 000)
   (H,  4, 1.44,  3, 126.00,  2, 170)
  )
   ReturnConnector()
   Connector()
   allowed_groups()
)
(
```

```
  Z(#9
    (N, −3, 0.7, −2, 000.00, −1, 000)
    (C,  0, 1.5, −3, 126.00, −2, 170)
    (C,  1, 1.5,  0, 108.00, −3, 180)
    (C,  2, 1.5,  1, 108.00,  0, 000)
    (C,  3, 1.5,  2, 108.00,  1, 000)
    (O,  1, 1.4,  2, 126.00,  0, 180)
    (O,  4, 1.4,  3, 126.00,  0, 180)
    (H,  2, 1.1,  1, 126.00,  0, 180)
    (H,  3, 1.1,  4, 126.00,  0, 180)
  )
    ReturnConnector()
    Connector()
    allowed_groups()
 )
)
```

List of Symbols, Abbreviations, and Acronyms

CCS	chosen chemical subspace
CJ	Chapman-Jouguet
HMX	octahydro-1,3,5,7-tetranitro-1,3,5,7-tetrazocine
IMD	inverse molecular design
LCAP	linear combination of atomic potentials
RDX	1,3,5-trinitroperhydro-1,3,5-triazine
TED	total energy of detonation
SMILES	simplified molecular-input line-entry system identifiers
VP-DFT	variation-of-particles density-functional-theoretical

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
1 (PDF)	DIR USARL RDRL WMM G J ORLICKI

INTENTIONALLY LEFT BLANK.