Published in final edited form as:
J Chem Inf Model. 2012 December 21; 52(12): 3144–3154. doi:10.1021/ci300363c.

Automation of the CHARMM General Force Field (CGenFF) I: bond perception and atom typing

K. Vanommeslaeghe and **A. D. MacKerell Jr**[*]
Department of Pharmaceutical Sciences, School of Pharmacy, University of Maryland, Baltimore, Maryland 21201

Abstract

Molecular mechanics force fields are widely used in computer-aided drug design for the study of drug-like molecules alone or interacting with biological systems. In simulations involving biological macromolecules, the biological part is typically represented by a specialized biomolecular force field, while the drug is represented by a matching general (organic) force field. In order to apply these general force fields to an arbitrary drug-like molecule, functionality for assignment of atom types, parameters and charges is required. In the present article, which is part I of a series of two, we present the algorithms for bond perception and atom typing for the CHARMM General Force Field (CGenFF). The CGenFF atom typer first associates attributes to the atoms and bonds in a molecule, such as valence, bond order, and ring membership among others. Of note are a number of features that are specifically required for CGenFF. This information is then used by the atom typing routine to assign CGenFF atom types based on a programmable decision tree. This allows for straightforward implementation of CGenFF's complicated atom typing rules and for equally straightforward updating of the atom typing scheme as the force field grows. The presented atom typer was validated by assigning correct atom types on 477 model compounds including in the training set as well as 126 test-set molecules that were constructed to specifically verify its different components. The program may be utilized via an online implementation at https://www.paramchem.org/.

Introduction

Molecular Mechanics (MM) is the method of choice for computational studies of biomolecular systems, owing to its modest computational cost. This makes it possible to routinely perform molecular dynamics (MD) simulations on biologically relevant systems that involve tens of thousands to hundreds of thousands atoms for simulation times from nanoseconds to microseconds. Even these limits are being pushed with recent work advancing towards millisecond time scales[1] and studies of micrometer-scale structures such as virus particles.[2] In the field of computer-aided drug design, MM methods are well established in the context of both ligand-based and structure-based ligand optimization approaches.[3] An essential component of MM and MD studies, both for biomolecular systems and for drug-like molecules and other types of ligands, is the force field used in the calculations. Indeed, it is the force field – in combination with the sampling of the relevant conformations – that dictates the accuracy of the obtained results. To enable accurate MM-based studies, a number of specialized empirical force fields for biological molecules[4–7] are available, often along with corresponding general force fields for drug-like molecules[8–11].

[*] To whom correspondence should be addressed: alex@outerbanks.umaryland.edu.

Supporting Information
An archive containing the "special case" model compounds designed for testing the atom typer is available as Supporting Information. This material is available free of charge via the Internet at http://pubs.acs.org.

Of note is the AMBER biomolecular force field[5] with its corresponding General Amber Force Field (GAFF),[9] because it was the first academic biomolecular force field for which a toolkit for automatic atom typing, named Antechamber, was made available.[12] The current work accomplishes the same goal for the CHARMM additive biomolecular force field[4] and the corresponding CHARMM General Force Field (CGenFF). CHARMM is a well-established force field for MD studies of biomolecular systems, including recent and explicitly optimized parameter sets for proteins,[13] nucleic acids,[14, 15] lipids[16] and carbohydrates.[17] The corresponding CHARMM General Force Field (CGenFF), which is the main focus of the current work, extends its scope to drug-like molecules, opening the door for applications in computer-aided drug design using the aforementioned methods.[8] Since its publication, CGenFF has been used for a variety of purposes,[18–25] and its coverage of medicinally relevant chemical space has expanded rapidly. CGenFF uses the CHARMM additive potential energy function to calculate the energy as a function of the Cartesian coordinates of the system, as shown in equation 1.

Intramolecular (internal, bonded terms)

$$\sum_{bonds} K_b(b-b_0)^2+\sum_{angles} K_\theta(\theta-\theta_0)^2+\sum_{dihedrals} K_\phi\,(1+\cos\,(n\phi-\delta))+\sum_{\substack{improper\\dihedrals}} K_\varphi(\varphi-\varphi_0)^2+\sum_{Urey-Bradley} K_{UB}(r_{1,3}-r_{1,3;0})^2 \qquad (1)$$

Intermolecular (external, nonbonded terms)

$$\sum_{nonbonded}\frac{q_iq_j}{4\pi Dr_{ij}}+\varepsilon_{ij}\left[\left(\frac{R_{\min,ij}}{r_{ij}}\right)^{12}-2\left(\frac{R_{\min,ij}}{r_{ij}}\right)^{6}\right]$$

The bonded or intramolecular part of the potential energy function consists of terms for the bonds, valence angles, torsion or dihedral angles, improper dihedral angles and a Urey-Bradley term, where b_0, Θ_0, φ_0 and $r_{1,3;0}$, respectively, are the bond, angle, improper and Urey-Bradley equilibrium values, the K's are the corresponding force constants, and n and δ are the dihedral multiplicity and phase. The nonbonded or intermolecular portion consists of an electrostatic term, with q_i and q_j being the respective partial atomic charges on atoms i and j, and a van der Waals (vdW) term, which is treated by the Lennard-Jones (LJ) 6–12 potential in which ε_{ij} is the well depth, $R_{\min,ij}$ is the radius, and r_{ij} is the distance between i and j. In addition, the energy function in equation 1 has been extended to include a 2D dihedral energy correction map, referred to as CMAP, which has been applied to improve the conformational properties of specific critical parts of biomolecules such as the ϕ, ψ terms in the peptide backbone,[26, 27] but is currently not in use in CGenFF. More details of the CHARMM potential energy function may be obtained from reference [28].

It is apparent from equation 1 that a simulation on any system of practical interest requires large numbers of parameters. To make the assignment of these parameters practical, force fields require atom types to be assigned to all the atoms in the system, with the parameters associated with combinations of atom types. For instance, the parameter list will contain K_ϕ, n and δ values for the dihedral parameters associated with all combinations of four atom types that occur in the molecules supported by the force field. Thus, the first step of assigning parameters for a chemical system is assigning atom types to that system. For biopolymers, this is trivial; a fixed set of atom types is defined for each monomer (ie. amino acid residue in the case of proteins, nucleotide in the case of nucleic acids or monosaccharide in the case of carbohydrates) and these atom types are applied to the polymer in accordance with the primary structure. Unfortunately, for general force fields for organic molecules, this approach cannot be followed and assigning atom types becomes more complicated. This is mainly due to the various ways in which basic functional groups can be combined, which requires different atom types to be assigned to the same basic functional group in different chemical environments. Most existing approaches for the assignment of atom types calculate basic properties related to the atomic connectivity for each atom and bond, and apply rules that stipulate which atom type should be assigned for which combination of these basic properties.[12, 29, 30] The present article describes in detail how this is done for CGenFF. Part II of this series will focus on the assignment of parameters and charges by analogy to molecules that were not explicitly parameterized during the development of CGenFF.[31]

Approach and algorithms

In the present work, atom typing is broken down into several steps, the first of which is to obtain information on the atoms in a molecule (eg. C, N, O, H etc.), the bonding pattern of those atoms and the types of bonds between those atoms from the input file (in mol2 format). This information is then used to identify and categorize ring systems in the molecule, a process that involves resolving “aromatic” bonds into a single resonance structure. Optionally, the same algorithm may be used to guess a molecule’s bond types based on its connectivity. Finally, all the above atom and bond properties are used in combination with rules to assign atom types to all the atoms in the molecule of interest. At this stage, it is possible to assign parameters to the molecule. However, parameter assignment on general organic molecules is quite complex due to the nearly infinite number of possible chemical connectivities that can occur. Therefore, this procedure is beyond the scope of the present work and will be presented in a second manuscript.[31]

Ring perception

Rings in the present work are defined as those rings with less than 8 atoms; larger rings can be treated in the MM formalism as acyclic chemical moieties because they have minimal ring strain and are rarely aromatic in practical cases. The atom typing rules first require knowledge of the smallest ring of which each non-bridgehead atom is a member. For bridgehead atoms, knowledge is required of the three smallest rings of which the atom is a member. Existing Smallest Set of Smallest Rings (SSSR) algorithms cannot be used for this purpose because they cannot guarantee that for each atom, the three smallest rings are members of the smallest *set*.[32, 33] In these cases, a common alternative to SSSR is an exhaustive ring search. However, exhaustive ring searches inherently scale very poorly with the system size, and yield substantially more information than required for the present application. For this reason, a two-stage ring perception algorithm was implemented, based on published work.

In the first stage of ring perception, the molecule's connectivity is reduced to a Homeomorphically Reduced Graph (HRG) in a slight variation of the procedure originally described by Balaban *et al.*[34] The present algorithm can be summarized by the following pseudocode:

```
while (the graph changed) {
  for each node {
    if (connectivity 0 or 1) remove node;
    else if (connectivity 2) {
      remove node;
      if (2 neigbors are the same) store ring;
else if (new edge not too long) connect 2 neighbors;
} } }
```

Where all atoms are represented by nodes or vertices, all of which are repeatedly evaluated and transformed until no more changes in the graph occur. Specifically, each node or vertex with connectivity 0 or 1 (ie. atom only covalently bonded to zero or one other atom) in the molecular graph is removed; in the same inner loop, each node with connectivity 2 is replaced by an edge (generalized bond) that contains a list of atoms that it replaces. The removal of nodes with connectivity 1 and replacement of nodes with connectivity 2 are respectively represented by the first and second transformation in Figure 1, although in reality, the algorithm does not wait until all nodes with connectivity 1 are removed before starting to replace nodes with connectivity 2. It should be noted that removal of nodes changes the connectivity of other nodes; for example, the first iteration ignores atom o because it has a connectivity of 3, then removes atoms p and q. In the second iteration, atom o's connectivity has changed to 1, so it is also removed, resulting in the second structure in Figure 1. To improve performance, edges in the graph are removed when they become longer than a preset number of atoms, which is a function of the largest ring size we are interested in identifying (7-membered in the case of CGenFF). Note that this edge removal occurs while replacing nodes with connectivity 2, but may again give rise to new nodes with connectivity 1 that need to be removed, which is the reason why the transformations of nodes with connectivity 1 and 2 happen within the same inner loop. Finally, whenever an atom is connected to itself by a single edge, indicating that this atom is part of an isolated, non-fused ring, the ring information is stored and the edge is removed from the graph, which is depicted by the third transformation in Figure 1. Again, this often gives rise to new nodes with connectivity 1 or 2. It should be reiterated that this figure is merely a pictorial representation; as shown above, there is only one loop that repeatedly evaluates all nodes in the graph and that may perform any of the transformations in the figure in any order until no more changes in the graph occur. Although this implies that the order in which transformations are carried out is dependent on the original order of the atoms, the final graph is always the same, containing only information on connected, fused ring systems. To summarize, the presented algorithm differs from Balaban's algorithm in the following respects:

1. Edges that are "too long" and isolated, non-fused ring are removed.
2. Changes in connectivity associated with edge removal are supported.
3. It does not attempt to identify rings beyond the isolated, non-fused cases.
4. The implementation attains enhanced performance at the cost of a slightly increased memory footprint. However, this is largely irrelevant given modern computer power and the presence of much slower steps in the atom typing program.

A logical next step to characterize the remaining rings (ie. the ones that were not already identified by the HRG algorithm) would be to proceed with Hanser et al's collapsing P-Graph algorithm,[35] but as discussed above, our goal is not to conduct an exhaustive ring search. Instead, the second stage of our program is a non-SSSR variant of Figueras's breath-first message-passing algorithm[36] (Figure 2). In this algorithm, a "message" is initiated at a selected "parent" atom in the graph that has a connectivity of 3 or more (atom i in the figure), and iteratively produces "child" messages on all adjacent atoms, where each "child" message carries the sequence of atoms that describes its propagation from the parent atom. Whenever a collision occurs (ie. a message tries to propagate to an atom that is already occupied by another message), a ring closure is found and the list of atoms in the ring can readily be extracted from the sequences contained in the two colliding messages. Our core message-passing algorithm differs from Figueras's algorithm in the following aspects:

1. Since the HRG allows for multi-bond edges, a provision was added for messages to spend multiple iterations traveling over a single edge.
2. In Figueras's algorithm, for the sake of performance, atoms are removed from the graph whenever they were no longer necessary for finding the SSSR. Also, "interfering" ring closures (ie. rings that could not be part of the SSSR) are ignored. As discussed above, the SSSR is not appropriate for our purpose; therefore, these two SSSR-related features are not present in our algorithm.
3. Messages are deleted after having traveled over 4 bonds, since we are only interested in rings smaller than n=8. This, together with the HRG reduction in the first stage, significantly limits the proliferation of messages, as well as imposing a hard limit on the algorithm's number of iterations.

This core message-passing algorithm is repeated initiating from each atom that was not eliminated by the HRG algorithm (ie. each atom with connectivity higher than 2) as a parent atom. This directly yields the desired ring memberships for the parent atom, as well as the smallest ring membership for the atoms with connectivity=2 on the edges attached to the parent atoms (which are implicitly present in our HRG as atom lists associated with edges). Although in analogy with Figueras's actual implementation, it might be possible to eliminate atoms from consideration, doing so would make the algorithm considerably more complex and would thereby introduce a risk of failure on non-trivial systems; indeed, Berger *et al.* identified cases where Figueras's algorithm fails to find the SSSR.[33] Our algorithm can be considered a trade-off between computational cost and algorithmic complexity.

Because the CGenFF atom typing scheme only considers 5-, 6- and 7-membered rings as aromatic, whenever a 3- or 4-membered ring is found, its nature is identified (see "ring typing and aromaticity perception" below) and this information is immediately applied to the atoms that are part of the ring. Non-duplicate 5-, 6- and 7-membered rings, on the other hand, are stored in an array because their nature depends on the outcome of the aromaticity perception, which can only be performed after all potential aromatic rings are identified. It should be noted that this CGenFF-specific behavior, including the limit on the ring size and number of rings, is controlled by compile-time constants, and can easily be modified if the need arises to apply the CGenFF atom typer to other force fields.

Resolution of resonance

Although bond orders do not explicitly occur in the class I potential energy function (equation 1), bond order information is implicitly present in the form of the atom types, and thus is required in order to assign these atom types correctly. Antechamber assumes that no bond order information is given by the user, and derives all bond order information from the connectivity. In contrast, the CGenFF atom typer allows the user to supply bond orders, because (1) it is easy to accidentally create a connectivity with a few missing hydrogen atoms, which may give rise to incorrect bond orders resulting in inappropriate conformational energetics and (2) we want to give expert users maximum control over the bond orders in their molecule. For example, if the user wishes to perform force field parametrization on a fragment of a larger molecule,[8] the bond orders need to be taken from the larger molecule, even if they are unphysical when considering the fragment as an isolated compound. Beyond this, some degree of bond type perception is needed because most molecular data formats that include bond order information allow for bonds to be marked as "aromatic", and most molecular modeling programs make use of this feature. It is also necessary to resolve consecutive aromatic bonds into a single resonance structure in order to correctly determine formal charges; for instance, the nitrogen atoms in the pyrrole and the pyridinium structures in Figure 3 have identical connectivities and bond orders, yet the nitrogen atom is positive in pyridinium but neutral in pyrrole. Moreover, the atom typing rules are based on a single resonance structure in order to eliminate ambiguity. Indeed, some but not all molecular modeling packages mark all bonds with bond order 1.5 as aromatic, including moieties such as positive amidinium, negative carboxylate, and neutral nitro groups. This becomes problematic when the symmetry is broken; for instance, one could devise an amidinium structure with an electron-donating substituent on one nitrogen atom and an electron-withdrawing substituent on the other nitrogen atom, causing one of the resonance structures to be strongly favored. In cases like this, it would be necessary to arbitrarily decide whether to apply two aromatic bonds or one single and one double bond. Indeed, the bond order of an amide bond may be considered 1 or 1.5, with different methods providing different answers. Another rationale for the implementation of bond type perception functionality is the widespread use of programs such as OpenBabel[37] that accept raw 3D coordinates and assign the bond orders based on the molecular geometry, which may give rise to the “aromatic” designation being assigned to bonds that are too short to fit the single bond criteria and too long for the double bond criteria. In this context, aromatic bonds can be regarded as bonds with an unknown bond order that can only be resolved based on the connectivity. However, it should be emphasized that assignment of bond orders based on 3D coordinates is not always guaranteed to be correct, regardless of whether it is followed by connectivity-based bond type perception or not. Specifically, if the initial 3D coordinates are heavily distorted and hydrogen atoms are absent, as is commonly the case for small molecules in macromolecular X-ray structures,[38] no amount of automatic or manual processing can unambiguously resolve the chemical structure. Therefore, it is strongly recommend that chemical structures of small molecules *not* be extracted from protein X-ray structures. Rather, these chemical structures should be verified in the primary literature.

Like Antechamber, the present atom typer uses a penalty score to rank resonance structures. However, unlike Antechamber, the penalty score is based on the formal charge and aromaticity of the molecule, rather than on atomic valences. Specifically, the penalty score is given by equation 2, where $Z_{molecule}$ is the molecule's total charge, Z_{atom} is the atom’s formal charge, $N_{rings}(5\text{–}7,sp^2)$ is the number of “potential aromatic rings” and $N_{rings}(arom)$ is the number of aromatic rings. In this context, potential aromatic rings are defined as 5, 6 or 7-membered rings that contain only atoms with 3 or fewer substituents. The coefficients in equation 2 are empirical. They were initially chosen based on the observation that a resonance structure that results in a net charge is less likely to be correct than a zwitterionic resonance structure, which in turn is often less favorable than a resonance structure in which a potential aromaticity is broken. While beta-testing the original formula resulting from the above logic, it was found to perform remarkably well, except for a few rare cases where the algorithm made arbitrary decisions between a negative phenolate and a positive ammonium group; structures **H** and **I** in Figure 4 represent a typical case. In line with this example, the negative phenolate was unlikely in the majority of these cases, so we changed the coefficients from 4, 2, 2, 1 to 8, 4, 3, 2, respectively, which gave satisfactory results on nearly all molecules tested. The rare exceptions were nontrivial and could not easily be codified in a simple mathematical rule.

$$penalty=8\,|Z_{molecule}|+4\sum_{\substack{atoms\\ with\ Z<0}}|Z_{atom}|+3\sum_{\substack{atoms\\ with\ Z>0}}|Z_{atom}|+2\left(N_{rings}\left(5-7,sp^2\right)-N_{rings}(arom)\right)\qquad(2)$$

For the purpose of atom typing, the resonance structure with the lowest penalty is used, as determined by the (formally recursive) “resolution of resonance” algorithm. This algorithm operates only on the subgraph of aromatic bonds; therefore, the words “edge” and “node” in the following discussion respectively mean “aromatic bond” and “atom with one or more aromatic bonds”. Initially, all edges are marked “untraversed”, and the graph is traversed in a depth-first fashion, starting at an arbitrary node.

1. If the node has more than one untraversed edge, the algorithm arbitrarily chooses one, and this choice is recorded in a stack-like data structure that will henceforward be referred to as “the trail”.

2. If the bond order of the chosen edge is ambiguous,[*] an arbitrary bond order is chosen and this choice is again recorded in the trail. Also, the edge will be marked with the bond order and thus becomes “traversed”.

3. If the next node represents a nitrogen atom, a formal charge (0 or +1) is arbitrarily chosen, and this choice is recorded in the trail. Similarly, terminal oxygen atoms (ie. oxygen atoms bound to only one atom) can either be neutral or negatively charged.

4. If the next node has one or more untraversed edges, the procedure is repeated from step (1); if the node has no untraversed edges, (as when visiting an intersection for the second time or arriving at a node where the resonance chain ends), the valence of the node is verified.

   - If the valence is violated, the trail is traversed backwards, erasing entries from the trail and marking edges as “untraversed” at every step. This process continues until either a choice of formal charge or a choice of bond order is encountered. The choice is changed, and the process is resumed from step (1). In the following discussion, this procedure will be referred to as “destructive backtracking”. Note that choices have a certain hierarchy so that the same exact combination of choices can never be considered more than once.

   - If the valence is satisfied, the last node where a direction was arbitrarily chosen is identified by searching backwards through the trail. The algorithm jumps to this node, recording the jump in the trail *after* the path that led to the jumping point, so that the trail can later be used to assign bond orders to this branch of the molecule. Then, it continues from step (1) down the previously untraversed node. This procedure will henceforward be referred to as “constructive backtracking” If no node can be found to jump to, a valid resonance structure is found. It is checked whether this resonance structure has a lower penalty score than the best resonance structure found so far. If yes, the current resonance structure is stored as “best resonance structure”, together with its penalty score. If this score is 0, the algorithm exits because the scoring system does not allow for better resonance structures with negative scores. Otherwise, the above procedure for a violated valence is followed to allow the algorithm to find other resonance structures.

This procedure can best be illustrated by the example in Figure 4. Starting from C_1 in the aromatic subgraph **A**, the algorithm establishes a double bond to C_2, and from there, a single bond to C_3, yielding state **B**. At this point, the algorithm makes the arbitrary decision to assign a double bond to C_3–C_4. After traversing this bond, it continues traversing the ring starting with C_4–C_5, arbitrarily assigning a double bond to C_7–C_8. When arriving back at

[*] This is not necessarily the case; for example, if a carbon atom has a double bond outside of the “aromatic subgraph” and two edges representing aromatic bonds, both edges must necessarily represent single bonds.

C_4, the valence turns out to be violated, triggering destructive backtracking to C_7 (state **C**). This time, a single bond is assigned to C_7–C_8. As a result, C_4's valence is now satisfied and the algorithm performs constructive backtracking, jumping to C_7 and continuing its traversal towards O_{12}. Also depicted in state **D** is the constructive backtracking that results in a jump from O_{12} to C_3. The end result is state **E**, where the valence of C_{10} turns out to be violated, triggering destructive backtracking to the last remaining point of choice, C_3. The algorithm now chooses a single bond for C_3–C_4, and continues traversing the graph in the same order as before. This time, a choice of ionization state exists when arriving at N_6 (state **F**); the algorithm chooses the neutral state and therefore continues with a single N_6-C_7 bond. Just as in state **C**, the valence at C_4 is violated in state **G** and destructive backtracking to C_7 is triggered. This is followed by two constructive backtracks that are not shown in the figure but are analogous to the ones depicted in state **D**. This results in state **H**, where a valid resonance structure is found. The penalty score for this structure is 12 (8 for the total charge + 4 for the negatively charged atom; the ring satisfies Huckel's rule and is therefore considered aromatic). As this is the lowest penalty score found so far, the current trail is saved as "best trail" (not shown). Then, destructive backtracking towards the last point of choice N_6 is triggered. This time, N_6 is protonated, which ultimately results in valid resonance structure **I**. This state has a penalty score of 11 because a positive charge has a slightly lower penalty than a negative charge as discussed above. As this is lower than the previous lowest score the "best trail" is replaced with the current trail and the ensuing destructive backtracking makes the algorithm jump to C_4, where it chooses to assign a single bond to C_4-C_5. This results in valid resonance structure **J**. As the penalty score is not lower than the previous lowest score, this state is discarded and destructive backtracking is triggered one last time, to C_7. The algorithm this time assigns a single bond to C_7-C_8, which results in a valence violation at C_4 (not shown). No more choices exist at this point, so the best trail, which corresponds to state **I**, is applied to the molecule and passed on to the ring typing and aromaticity perception routine.

The way this algorithm traverses the molecular graph is similar to Ray and Kirsch's basic backtracking algorithm,[39] although its application is different (resolution of resonance as opposed to subgraph isomorphism). It also exhibits the same unfavorable factorial scaling. For drug-like molecules, this is rarely a problem because the resonance chains in such molecules are generally small; however, processing a large graphene or fullerene in which all bonds are marked aromatic would cost a substantial amount of computer time. It should be noted that, while our code is algorithmically different from Antechamber's bond type perception program, it offers similar functionality and, likewise, can be used for assigning bond orders when none are supplied by the user. Indeed, since accurate bond orders are not always available (especially in high-throughput applications), the CGenFF atom typer does offer an option to discard the input bond orders and assign them internally instead. However, as discussed at the beginning of this section, the process of assigning bond orders sometimes involves arbitrary decisions and critically depends on having the correct number of hydrogens in the correct protonation and tautomeric states, so routine usage of this option is not recommended. Finally, it should be noted that minor enhancements to the current code would enable calculating fractional bond orders as the average of the formal bond orders of all the resonance structures, weighted by a function of their respective penalties, which could potentially be used for assigning bonded parameters in future force fields.

Ring typing and aromaticity perception

The CGenFF atom typing rules require all rings to be classified in one of 4 classes:

- *All-sp^3* rings consisting only of atoms not participating in any double bonds

- *All-sp^2* rings consisting only of atoms engaged in a double bond, with the exception of at most one N, O, P or S atom that has only single bonds.
- *Aromatic* rings are 5, 6 or 7-membered rings that contain only atoms with 3 or fewer substituents and have 6 in-ring π-electrons (ie. following Huckel's rule), where
  - ○ an in-ring double or triple bond counts for 2 π-electrons
  - ○ any atom that participates in an out-of-ring double bond and is part of another aromatic ring counts for 1 π-electron
  - ○ a heteroatom that only participates in single bonds and is not part of another aromatic ring counts for 2 π-electrons
  - ○ a heteroatom that only participates in single bonds and is part of another aromatic ring may count for either 1 or 2 π-electrons, whichever makes the current ring aromatic

  Applying this last criterion requires an iterative aromaticity perception routine. Specifically, in every iteration all potential aromatic rings are verified for aromaticity by electron counting. If any of these rings changed status during this iteration, another iteration is required.

  The motivation for using this specific definition of aromaticity is purely pragmatic, as it was found that this simple set of rules exactly reproduces the occurrence of aromatic and non-aromatic atom types in all 315 cyclic model compounds that were manually typed and parametrized during the CGenFF parametrization. However, the fact that the atom types on these compounds were chosen to reproduce physically relevant target data implies that our aromaticity perception rule is physically relevant to at least some extent.
- *Mixed sp^2/sp^3* rings are rings that do not belong in any of the above classes

Once the type is established for each ring, this information is applied to all the atoms in the ring. Bonds simply are marked as “endocyclic” or “exocyclic/acyclic”. This information is necessary and sufficient for CGenFF atom typing but might not be enough for other force fields. The choice of ring information available to the atom typer and the treatment of aromaticity are the only parts of the program that are specific to CGenFF and would require modifications in order to be applicable to other force fields.

Atom typing

In most existing atom typers, a rule for assigning a specific atom type fully contains all properties that need to be matched in order to assign that atom type; in the following discussion, we will call this a “one-rule-per-atom-type scheme”. In contrast, our atom typer is a programmable decision tree, using a “language” that is described in the present paragraph. The atom typer's program (aka. “rule file”) contains a number of *categories*, each of which starts with a “cat” statement defining the name of the category, and ends with an “end” statement (see Figure 5 for a schematic breakdown of part of the rule file). Between these two statements, one or more *rules* are contained. A rule starts with an *action*, followed by a colon and zero or more *conditions*, and ends with zero or more *optional actions*. Table 1 gives an overview of all the conditions and optional actions supported by the atom typer. The action can either be assigning an atom type (action keyword “typ”, followed by the atom type), or jumping to a “subcategory” (action type “sub” followed by the name of the subcategory). *For each atom in the input molecule*, the atom typer starts at the root of the tree (category “main”; see Figure 5), evaluating all the rules in this category. When a rule is found for which all the conditions are true, its associated action in carried out (along with any optional actions), resulting in the evaluation of increasingly specific subcategories until a “typ” action is encountered. Optional actions are mainly used to assign special attributes to atoms, such as a nonzero formal charge, an improper dihedral, or a warning message (respective keywords “charge …”, “impr”, and “warn …”). It should be noted that the formal charge that is assigned through the rule file and will be used as a basis for partial charge assignment is independent of the formal charge used in the “resolution of resonance” routine discussed above. This second formal charge assignment is necessary to obtain a symmetric charge distribution in delocalized cases like amidinium, guanidinium and carboxylate. Indeed, in the example of the carboxylate group, the resolution of resonance routine would assign a -1 formal charge to one of the oxygen atoms, while the rules in the rule file place the -1 formal charge on the carbon atom instead, and leave it up to the partial charge assignment algorithm (discussed in the accompanying part II article) to distribute this charge properly over the molecule. Other optional actions are “err …”, which causes the atom typer to exit immediately with an error message, and “altnum”, which causes a question mark in the atom type to be substituted by a “1” or a “2” such that in a chain of conjugated double bonds, double bonded atoms get the same symbol and single-bonded atoms get a different symbol. This is a well-established way to obtain sensible bond lengths and dihedral parameters in chains of conjugated double bonds. The current implementation, using only two such atom types and considering only 5, 6 and 7-membered rings aromatic, will fail on [4N+10]annulenes (with N>=0). Fortunately, such rings are rare in drug-like molecules because of synthetic difficulty and undesirable pharmacological properties.[†] If necessary, the algorithm could be extended to allow for a third conjugated double bond atom type, but this has not been done to date because it would result in a combinatorial explosion in the number of conjugated double bond parameters in the force field.

Most *conditions* operate on atoms:

- “el …” matches a specified element
- “elha” matches all halogens and “elos” matches oxygen and sulphur. In the future, if CGenFF is extended to support selenium, this will require a decision as to whether to include this element in the elos category. This decision will be based on the similarity and atom type-compatibility between representative organoselenium compounds and their sulphur and oxygen-containing equivalents.
- “nb …” stands for the number of electrons involved in bonds to other atoms, more informally referred to as the valence.
- “rings …” specifies the number of rings the atom should be a member of
- “ring3 …”, ring2 …”, “arom …”, “ring23 …” and “ring …” respectively match atoms that are a member of an a*ll-sp*^{3} ring, an a*ll-sp*^{2} ring, an *aromatic* ring, a *mixed sp*^{2}*/sp*^{3} ring, and any ring. For each of these conditions, the size of the ring must be specified. Once a ring has been matched by one of these conditions, it is excluded from being matched by a subsequent condition; for example, “arom 6 arom 6” will match the bridging carbon atoms in naphthalene, but none of the other carbon atoms.

[†] Specifically, [10]annulene is unstable and [14]annulene is reactive, so the smallest stable annulene that cannot be successfully typed is [18]annulene, and such large annulenes are rare in drug-like molecules because of synthetic difficulty and undesirable pharmacological properties. Although [10]annulene and [14]annulene can be stabilized by bridging the ring, this will generally give rise to 5- or 6-membered rings to which different atom types apply, thus circumventing the problem. An exception is 1,6-methano[10]annulene, but this molecule exhibits a special homoconjugative transannular interaction which may not be described well by the force field under any circumstance.

- "self" matches the parent atom that is being typed. When used in rules with 3 or more nested "ne" conditions (see below), this makes it possible to explicitly detect ring closures, and thereby provides finer-grained ring topology information in polycyclic molecules.

The other (ie. non-atomic) conditions are:

- "ne" (short for "neighbor"), followed by one or more series of conditions, each of which are enclosed in round brackets. The first series of conditions is applied to all the neighboring atoms of the parent atom. As soon as a matching neighbor is found, that neighbor is marked as "used", and the next series of conditions is applied to all the neighboring atoms that are still "unused". This process is repeated until either all series of conditions have been matched, which implies the "ne" condition is satisfied, or no match is found for a series of conditions, which implies the "ne" condition is not satisfied. "ne" statements can be nested in order to traverse several bonds. It is important to emphasize that once a condition is satisfied by a neighboring atom, that atom cannot be matched by any subsequent conditions within the same rule. For example, if an atom has substituents A and B, and a rule is applied to it that contains "ne" followed by two conditions, the first matching both A and B, and the second only matching A, it is possible that the first condition will match and "consume" A, leaving the second condition with no matching substituent and causing the rule to be rejected. Conversely, if the order of the atoms is different, it is equally possible that the first condition will match B and the second one A, satisfying the rule. The programmer of the rule file should be aware of this property and actively avoid the above undefined behavior, mainly by putting more specific rules first. In practice, this proved relatively straightforward to do while implementing the CGenFF atom typing rules. The main advantage of this approach is a substantial decrease in algorithmic and computational complexity; allowing out-of-order matching would be equivalent to determining subgraph isomorphism, which is an NP-complete problem.
- "bo …" matched the bond order of the last bond traveled by a preceding "ne" condition (and thus cannot be applied to the parent atom).
- "inring" is satisfied if the last bond traveled by a preceding "ne" condition is part of a ring.
- An exclamation mark, followed by a series of conditions enclosed in round brackets, acts as a "not" operator.
- "or", followed by one or more series of conditions, each of which are enclosed in round brackets, acts as an "or" operator.

The additional functionality in our programmable decision tree, compared to more conventional one-rule-per-atom-type implementations, was deemed opportune because of the *ad hoc* nature of the CGenFF atom types. Indeed, CGenFF parametrization did not start from a predetermined atom typing scheme; rather, it started with a basic palette of atom types derived from the biomolecular CHARMM additive force field, and consistent atom typing rules were created in parallel with the parametrization of model compounds, driven by how well the atom types on a model compound could reproduce the target data. Where necessary, new atom types were introduced in an empirical fashion. As a consequence, the current atom typing scheme uses very generic atom types for some functional groups, while having very specialized atom types for other moieties. Decisions concerning the creation of new atoms types were made empirically depending on the need as it arose during the parametrization rather than on predetermined chemical concepts. We see this as one of

CGenFF's unique strengths, as it gave us the freedom to adjust our atom typing scheme whenever our chemical expectations were disproven by the target data during the parametrization. The disadvantage to this approach is a considerable increase in the complexity of the atom typing rules; some of the rules for our more specialized atom types would become exceedingly long and nontransparent in a one-rule-per-atom-type scheme. Another motivation for using a decision tree is that CGenFF is still actively being refined towards wider and more accurate coverage of chemical space. This will require frequent adjustment of the rules, which is anticipated to be easier to accomplish with a programmable decision tree rather than with a one-rule-per-atom-type scheme. Finally, it should be noted that a decision tree is expected to be computationally more efficient than a one-rule-per-atom-type scheme. In the latter scenario, the linear list of rules must be evaluated in a fixed order until the correct atom type is found, while with a decision tree, the atom is classified in increasingly specific categories, and no rules outside these categories are evaluated.

Results and Discussion

Case study 1: the pyridinium ion

As a simple case study, the assignment of atom types on the aromatic pyridinium ion (rightmost structure in Figure 3) is considered. The 6-memberd ring is identified in the first stage of our ring perception algorithm. Specifically, in the first cycle of the HRG algorithm (corresponding to the first transformation in Figure 1), all hydrogen atoms are removed from the molecular graph. Subsequently, atoms in the ring are replaced by multi-atom edges (middle transformation in Figure 1), until the algorithm eventually detects that it is trying to connect an atom to itself. At this stage, the 6-membered ring is stored in an array of rings to be submitted to the aromaticity perception algorithm, and all but one atom is removed from the graph (last transformation in Figure 1). As the molecule in this case is a single ring, the remaining atom has a connectivity of 0, and is thus removed from the graph in the next cycle, leaving 0 atoms to be submitted to stage 2 (the breath-first message-passing algorithm). Next, assuming that the bonds in the ring were defined as aromatic in the input structure, the resolution of resonance algorithm is called. In the case of pyridinium, it will only find 2 valid resonance structures, both with a penalty score of 6 (4 for the net +1 charge plus 2 for the +1 charge on the nitrogen atom) and arbitrarily pick one (just like in benzene, both resonance structures are equivalent). Finally, the algorithm for ring typing and aromaticity perception is passed a 6-membered ring with 3 in-ring double bonds, which it identifies as aromatic. This information is applied to the 6 endocyclic atoms and bonds, and the resulting molecular description is passed on to the atom typer.

For the sake of brevity, we will only discuss the typing of the nitrogen atom in this case study; the other atoms are analogous. For any given atom, the atom typer starts off at the root of the decision tree, in category “main” (Figure 5). The first matching rule in this category reads “sub NG : el N”; since the element is N, the atom typer jumps to the subcategory NG (ie. “nitrogen general”). All rules in this subcategory are meant to match special unsupported cases and produce warning or error messages, except for the last rule, which simply reads “sub NG_ :”.

Since there are no conditions in this rule, it is automatically matched, and the atom typer proceeds to “cat NG_”. There, it matches the rule “sub NG_2 : ne (bo 2)” because it is double bonded to one of its neighbors, so the atom typer jumps to the subcategory of sp^2 nitrogen atoms named NG_2. Finally, this category contains a rule “typ NG2R61 : arom 6 ne (bo 1) (bo 1) charge 1”, the condition of which is satisfied because the atom is part of an aromatic 6-memberd ring and single bonded to two of its neighbors. Therefore, it is assigned a formal charge of +1 and an atom type of NG2R61. It should be noted that optional properties such as the formal charge do not necessarily need to be applied by the same rule that assigns the atom type; for example, the aforementioned category NG_ contains a rule “sub NG_3P : nb 4 charge 1” that assigns a positive charge to nitrogen atoms with valence=4 but leaves the assignment of a specific atom typing to the subcategory NG_3P.

Case study 2: 1,3-dipentene

As a second case study, we will consider the *sp*^{2} carbon atoms in 1,3-dipentene (Figure 6), starting with C4. After jumping from the category “main” to the category “CG” (Figure 5), the first matching rule is “sub CG2 : ne (bo 2)” because C4 is double bonded to one of its neighbors. Consequently, the atom typer jumps to the subcategory CG2. This category, together with the category CG2_,[‡] acts as a “filter” that identifies some special cases and assigns improper dihedrals where needed. Specifically, the first rule in category CG2 only matches carbon dioxide, while the second rule matches bridging carbons between 2 aromatic rings and jumps directly to subcategory CG2__, skipping subcategory CG2_ (dashed arrow in Figure 5) and thereby suppressing the assignment of improper dihedrals. The third rule is specific for biphenyl moieties. Since none of these rules are triggered in the present case study, the fourth rule is automatically matched and the atom typer proceeds to subcategory CG2_. Although all rules in this category point to subcategory CG2__, the first three assign improper dihedrals, while the fourth, which is matched in the present case study, does not. Category CG2__ indentifies a variety of rings, out of which only the “typ CG2R71 : arom 7” 7-membered aromatic ring rule is shown in Figure 5 (this rule is matched in azulene and related compounds). It also has a “sub CG2O : ne (bo 2 elos)” rule for identifying carbonyl-type carbon atoms. The remaining three rules point to the subcategory CG2D; out of these 3, the first one assigns a positive charge (which must be localized on the carbon atom in order to maintain symmetry in amidinium and guanidinium groups) and the second one warns the user that enolate groups are poorly supported (in CGenFF version 2b7, which is current at the time of writing). Again, none of these rules are satisfied in the present case study, so the last rule is matched and the atom typer proceeds to the subcategory CG2D without further action. Finally, in the category CG2D for double bonded carbons, the rule “typ CG2DC? : ne (bo 2 ne (! (el H) bo 1 ne (bo 2))) altnum” is matched because the atom under consideration (C4) is double bonded to an atom (C3) that is singlebonded to a non-hydrogen atom (C2) that, in turn, is double bonded to a fourth atom (C1). Note that the same rule would also be applied when assigning an atom type to C1, except that that the rule “typ CG2DC3 : ne (el H) (el H) (ne (! (el H) bo 1 ne (bo 2)))” takes precedence because it occurs earlier in the list. This leads to C1 being assigned the specific atom type CG2DC3 for a CH2 carbon atom at the end of a chain of conjugated double bonds. The remaining *sp*^{2} carbon atoms C2 and C3 match the rule “typ CG2DC? : ne (! (el H) bo 1 ne (bo 2)) altnum”. As discussed above under the heading “Atom typing”, the “altnum” directive in these rules causes the question mark in C2, C3 and C4’s type CG2DC? to be substituted by a “1” or a “2” such that in a chain of conjugated double bonds, double bonded atoms get the same symbol and single-bonded atoms get a different symbol. Specifically, C2 may get assigned the type CG2DC1 while C3 and C4 become CG2DC2; accordingly, the bonded parameters are such that CG2DC1–CG2DC2 represents a single bond while CG2DC3–CG2DC1 and CG2DC2–CG2DC2 represent double bonds. It should be noted that depending on the input order of the atoms and bonds in the mol2 file, C2 might get assigned CG2DC2 while C3 and C4 become CG2DC1. However, this arbitrary choice does not influence the energetics because the types GC2DC1 and CG2DC2 are treated in a symmetric fashion in the CGenFF parameter file.

[‡] Note that the typing rules parser differentiates between single underscore, “_” and double underscore “__”.

Validation

When initially applying the algorithm described in the previous section to all 477 full model compounds in CGenFF,[8] a number of discrepancies were found. This allowed us to fix mistakes in our initial rule file, as well as occasional mistakes in the manual assignment of atom types on existing model compounds. Additionally, 126 "special case" molecules were constructed to specifically test the different functions of the atom typer, including non-trivial rules in the rule file; a full archive of these molecules is available in the Supporting Information. Manual analysis of the atom types generated by the CGenFF atom typer confirmed that all atoms in those molecules were typed correctly. Subsequently, 18 months of public beta testing involving more than 10000 model compounds brought relatively few flaws to light, which were duly corrected.

Summary

An atom typer was constructed for the CHARMM General Force Field. The main innovation in the CGenFF atom typer is the use of a programmable decision tree, which makes it straightforward to implement very complex atom typing rules and adjust these rules if the underlying force field is refined towards wider and more accurate coverage of chemical space. Criteria for assignment of bond orders and ring and aromaticity perception extend previous efforts with variations introduced to meet specific needs of CGenFF as well as for computational efficiency. The current atom typer and rule file are able to correctly type all existing model compounds in CGenFF as well as a number of "special cases" designed to test their limits. The CGenFF atom typer is a fast and flexible tool for computer-aided drug design projects, as well as a solid and convenient basis for future expansion of CGenFF. Access to the CGenFF program for automatic atom typing and assignment of parameters and charges by analogy is provided at https://www.paramchem.org/.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

Financial support from the NIH (GM51501, GM070855, CA107331), the NSF (CHE-0823198), the Waxman Foundation, and the University of Maryland Computer-Aided Drug Design Center are acknowledged. Additionally, this research was supported in part by the NSF through TeraGrid[40] resources provided by NCSA and PSC.

References

1. Lindorff-Larsen K, Piana S, Dror RO, Shaw DE. How Fast-Folding Proteins Fold. Science. 2011; 334(6055):517–520. [PubMed: 22034434]
2. Freddolino PL, Arkhipov AS, Larson SB, McPherson A, Schulten K. Molecular Dynamics Simulations of the Complete Satellite Tobacco Mosaic Virus. Structure. 2006; 14:437–449. [PubMed: 16531228]
3. Shim J, MacKerell AD Jr. Computational ligand-based rational design: Role of conformational sampling and force fields in model development. MedChemComm. 2011; 2(5):356–370. [PubMed: 21716805]
4. MacKerell, AD., Jr; Brooks, B.; Brooks, CL., III; Nilsson, L.; Roux, B.; Won, Y.; Karplus, M. CHARMM: The Energy Function and Its Paramerization with an Overview of the Program. In: Schleyer, PvR; Allinger, NL.; Clark, T.; Gasteiger, J.; Kollman, PA.; Schaefer, HF., III; Schreiner, PR., editors. Encyclopedia of Computational Chemistry. Vol. Vol. 1. Chichester: John Wiley & Sons; 1998. p. 271-277.
5. Cornell WD, Cieplak P, Bayly CI, Gould IR, Merz KM Jr, Ferguson DM, Spellmeyer DC, Fox T, Caldwell JW, Kollman PA. A Second Generation Force Field for the Simulation of Proteins,

Nucleic Acids, and Organic Molecules. Journal of the American Chemical Society. 1995; 117(19): 5179–5197.

6. Jorgensen WL, Maxwell DS, Tirado-Rives J. Development and Testing of the OPLS All-Atom Force Field on Conformational Energetics and Properties of Organic Liquids. Journal of the American Chemical Society. 1996; 118:11225–11236.

7. Oostenbrink C, Villa A, Mark AE, Van Gunsteren WF. A biomolecular force field based on the free enthalpy of hydration and solvation: The GROMOS force-field parameter sets 53A5 and 53A6. J. Comput. Chem. 2004; 25(13):1656–1676. [PubMed: 15264259]

8. Vanommeslaeghe K, Hatcher E, Acharya C, Kundu S, Zhong S, Shim J, Darian E, Guvench O, Lopes P, Vorobyov I, MacKerell AD Jr. CHARMM General Force Field (CGenFF): A Force Field for Drug-Like Molecules Compatible with the CHARMM All-Atom Additive Biological Force Fields. J. Comput. Chem. 2010; 31(4):671–690. [PubMed: 19575467]

9. Wang J, Wolf RM, Caldwell JW, Kollman PA, Case DA. Development and testing of a general AMBER force field. J. Comput. Chem. 2004; 25:1157–1174. [PubMed: 15116359]

10. Jorgensen WL, McDonald NA. Development of an all-atom force field for heterocycles. Properties of liquid pyridine and diazenes. Journal of Molecular Structure (Theochem). 1998; 424(1–2):145–155.

11. Price MLP, Ostrovsky D, Jorgensen WL. Gas-Phase and Liquid-State Properties of Esters, Nitriles, and Nitro Compounds with the OPLS-AA Force Field. J. Comput. Chem. 2001; 22:1340–1352.

12. Wang JM, Wang W, Kollman PA, Case DA. Automatic atom type and bond type perception in molecular mechanical calculations. Journal of Molecular Graphics and Modelling. 2006; 25(2): 247–260. [PubMed: 16458552]

13. Best RB, Zhu X, Shim J, Lopes PEM, Mittal J, Feig M, MacKerell AD Jr. Optimization of the Additive CHARMM All-Atom Protein Force Field Targeting Improved Sampling of the Backbone φ, ψ and Side-Chain χ_1 and χ_2 Dihedral Angles. Journal of Chemical Theory and Computation. 2012; 8(9):3257–3273.

14. Hart K, Foloppe N, Baker CM, Denning EJ, Nilsson L, MacKerell AD Jr. Optimization of the CHARMM Additive Force Field for DNA: Improved Treatment of the BI/BII Conformational Equilibrium. journal of Chemical Theory and Computation. 2012; 8(1):348–362. [PubMed: 22368531]

15. Denning EJ, Priyakumar UD, Nilsson L, MacKerell AD Jr. Impact of 2'-Hydroxyl Sampling on the Conformational Properties of RNA: Update of the CHARMM All-Atom Additive Force Field for RNA. J. Comput. Chem. 2011; 32(9):1929–1943. [PubMed: 21469161]

16. Klauda JB, Venable RM, Freites JA, O'Connor JW, Tobias DJ, Mondragon- Ramirez C, Vorobyov I, MacKerell AD Jr, Pastor RW. Update of the CHARMM All-Atom additive Force Field for Lipids: Validation on Six Lipid Types. Journal of Physical Chemistry B. 2010; 114(23):7830–7843.

17. Guvench O, Mallajosyula SS, Raman EP, Hatcher E, Vanommeslaeghe K, Foster TJ, Jamison FW II, MacKerell AD Jr. CHARMM Additive All-Atom Force Field for Carbohydrate Derivatives and Its Utility in Polysaccharide and Carbohydrate-Protein Modeling. Journal of Chemical Theory and Computation. 2011; 7(10):3162–3180. [PubMed: 22125473]

18. Rais R, Acharya C, Tririya G, MacKerell AD Jr, Polli JE. Molecular Switch Controlling the Binding of Anionic Bile Acid Conjugates to Human Apical Sodium-Dependent Bile Acid Transporter. Journal of Medicinal Chemistry. 2010; 53(12):4749–4760. [PubMed: 20504026]

19. Rais R, Acharya C, MacKerell AD Jr, Polli JE. Structural Determinants for Transport across the Intestinal Bile Acid Transporter Using C-24 Bile Acid Conjugates. Molecular Pharmaceutics. 2010; 7(6):2240–2254. [PubMed: 20939504]

20. Rosenbaum DM, Zhang C, Lyons JA, Holl R, Aragao D, Arlow DH, Rasmussen SG, Choi H-J, DeVree BT, Sunahara RK, Chae PS, Gellman SH, Dror RO, Shaw DE, Weis WI, Caffrey M, Gmeiner P, Kobilka BK. Structure and function of an irreversible agonist-β2 adrenoreceptor complex. Nature. 2011; 469(7329):236–242. [PubMed: 21228876]

21. Kruse AC, Hu J, Pan AC, Arlow DH, Rosenbaum DM, Rosemond E, Green HF, Liu T, Chae PS, Dror RO, Shaw DE, Weis WI, Wess J, Kobilka BK. Structure and dynamics of the M3 muscarinic acetylcholine receptor. Nature. 2012; 482(7386):552–556. [PubMed: 22358844]

22. Fribourg M, Moreno JL, Holloway T, Provasi D, Baki L, Mahajan R, Park G, Adney SK, Hatcher C, Eltit JM, Ruta JD, Albizu L, Li Z, Umali A, Shim J, Fabiato A, MacKerell AD Jr, Brezina V, Sealfon SC, Filizola M, González-Maeso J, Logothetis DE. Decoding the Signaling of a GPCR Heteromeric Complex Reveals a Unifying Mechanism of Action of Antipsychotic Drugs. Cell. 2011; 147(5):1011–1023. [PubMed: 22118459]

23. González A, Perez-Acle T, Pardo L, Deupi X. Molecular Basis of Ligand Dissocaition in β-Adrenergic Receptors. PLOS One. 2011; 6(9):e23815. [PubMed: 21915263]

24. Huang D, Caflisch A. The Free Energy Landscape of Small Molecule Unbinding. PLOS Computational Biology. 2011; 7(2):e1002002. [PubMed: 21390201]

25. Zhou K, Gao Y, Hoy JA, Mann FM, Honzatko RB, Peters RJ. Insights into Diterpene Cyclization from Structure of Bifunctional Abietadiene Synthase from *Abies grandis*. The Journal of Biological Chemistry. 2012; 287(9):6840–6850. [PubMed: 22219188]

26. MacKerell AD Jr, Feig M, Brooks CL III. Accurate treatment of protein backbone conformational energetics in empirical force fields. J Am Chem Soc. 2004; 126:698–699. [PubMed: 14733527]

27. MacKerell AD Jr, Feig M, Brooks CL III. Extending the treatment of backbone energetics in protein force fields: limitations of gas-phase quantum mechanics in reproducing protein conformational distributions in molecular dynamics simulations. J. Comput. Chem. 2004; 25:1400–1415. [PubMed: 15185334]

28. MacKerell AD Jr. Empirical Force Fields for Biological Macromolecules: Overview and Issues. J. Comput. Chem. 2004; 25:1584–1604. [PubMed: 15264253]

29. Halgren TA. Merck Molecular Force Field. I. Basis, Form, Scope, Parameterization, and Performance of MMFF94. J. Comput. Chem. 1996; 17(5–6):490–519.

30. Momany FA, Rone R. Validation of the General Purpose QUANTA 3.2/CHARMm Force Field. J. Comput. Chem. 1992; 13(7):888–900.

31. Vanommeslaeghe K, Raman EP, MacKerell AD Jr. Automation of the CHARMM General Force Field (CGenFF) II: assignment of bonded parameters and charges by analogy. in preparation. 2012

32. Section 12.7: Smallest Set of Smallest Rings (SSSR) considered Harmful. OEChem - C++ Theory Manual version 1.6.1, version 1.6.1 ed. Santa, Fe, New Mexico: OpenEye Scientific Software, Inc.; 2008. p. 59-60.

33. Berger F, Flamm C, Gleiss PM, Leydold J, Stadler PF. Conterexamples in Chemical Ring Perception. J. Chem. Inf. Comput. Sci. 2004; 44:323–331. [PubMed: 15032507]

34. Balaban AT, Filip P, Balaban T-S. Computer program for finding all possible cycles in graphs. J. Comput. Chem. 1985; 6(4):316–329.

35. Hanser T, Jauffret P, Kaufmann G. A New Algorithm for Exhaustive Ring Perception in a Molecular Graph. Journal of Chemical Information and Computer Sciences. 1996; 36(6):1146–1152.

36. Figueras J. Ring Perception using Breadth-First Search. Journal of Chemical Information and Computer Sciences. 1996; 36(5):986–991.

37. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR. Open Babel: An open chemical toolbox. Journal of Cheminformatics. 2011; 3:33. [PubMed: 21982300]

38. Liebeschuetz J, Hennemann J, Olsson T, Groom CR. The good, the bad and the twisted: a survey of ligand geometry in protein crystal structures. Journal of Computer-Aided Molecular Design. 2012; 26:169–183. [PubMed: 22246295]

39. Ray LC, Kirsch RA. Finding Chemical Records by Digital Computers. Science. 1957; 126(3278): 814–819. [PubMed: 17776535]

40. Catlett, C.; Allcock, WE.; Andrews, P.; Aydt, R.; Bair, R.; Balac, N.; Banister, B.; Barker, T.; Bartelt, M.; Beckman, P.; Berman, F.; Bertoline, G.; Blatecky, A.; Boisseau, J.; Bottum, J.; Brunett, J.; Bunn, J.; Butler, M.; Carver, D.; Cobb, J.; Cockerill, T.; Couvares, PF.; Dahan, M.; Diehl, D.; Dunning, T.; Foster, I.; Gaither, K.; Gannon, D.; Goasguen, S.; Grobe, M.; Hart, D.; Heinzel, D.; Hempel, C.; Huntoon, W.; Insley, J.; Jordan, C.; Judson, I.; Kamrath, A.; Karonis, N.; Kesselman, C.; Kovatch, P.; Lane, L.; Lathrop, S.; Levine, M.; Lifka, D.; Liming, L.; Livny, M.; Loft, R.; Marcusiu, D.; Marsteller, J.; Martin, S.; McCaulay, S.; McGee, J.; McGinnis, L.; McRobbie, M.; Messina, P.; Moore, R.; Moore, R.; Navarro, JP.; Nichols, J.; Papka, ME.; Pennington, R.; Pike, G.; Pool, J.; Reddy, R.; Reed, D.; Rimovsky, T.; Roberts, E.; Roskies, R.;

Sanielevici, S.; Scott, JR.; Shankar, A.; Sheddon, M.; Showerman, M.; Simmel, D.; Singer, A.; Skow, D.; Smallen, S.; Smith, W.; Song, C.; Stevens, R.; Stewart, C.; Stock, RB.; Stone, N.; Towns, J.; Urban, T.; Vildibill, M.; Walker, E.; Welch, V.; Wilkins-Diehr, N.; Williams, R.; Winkler, L.; Zhao, L.; Zimmerman, A. TeraGrid: Analysis of Organization, System Architecture, and Middleware Enabling New Types of Applications. In: Grandinetti, L., editor. High Performance Computing (HPC) and Grids in Action. Vol. Vol. 16. Amsterdam: IOS Press; 2007.

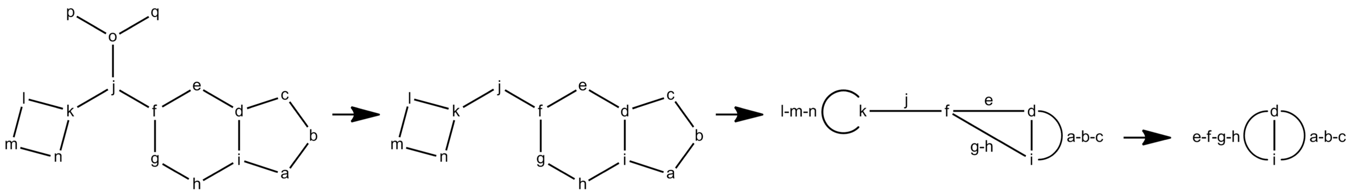


Figure 1.

Balaban’s HRG algorithm, and the further simplification in our variant.

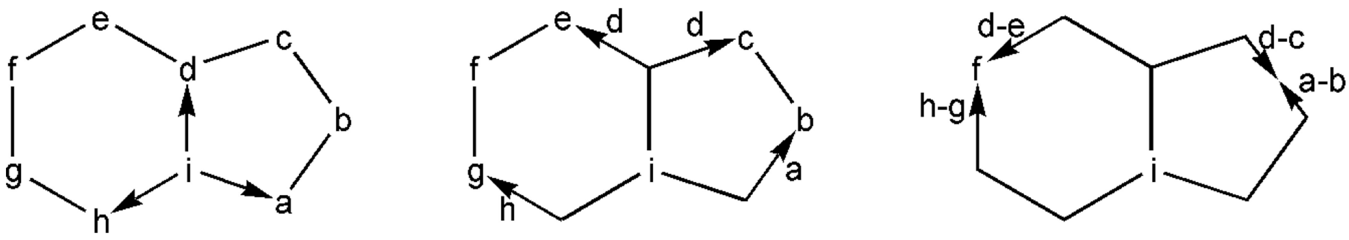


Figure 2.
Core breath-first message-passing algorithm, starting at a node with connectivity 3.

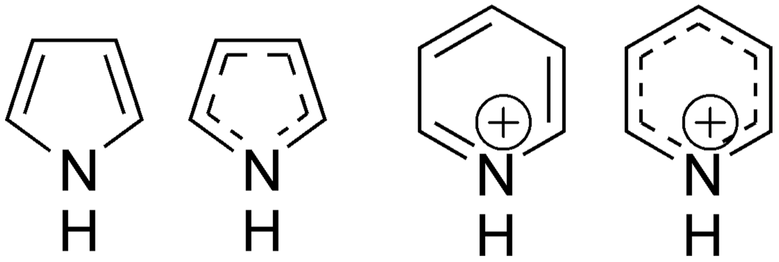
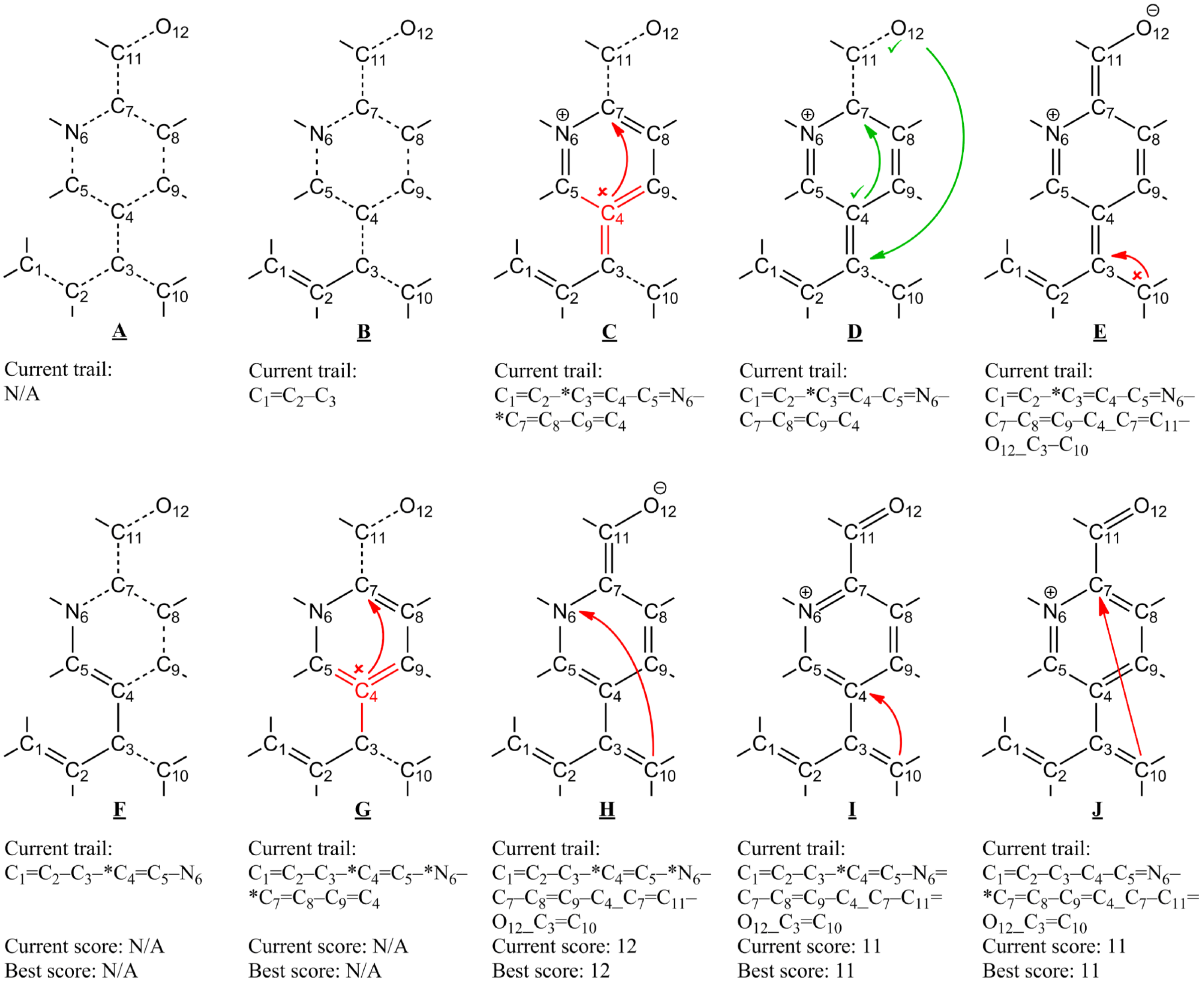


Figure 3.

Pyrrole and the pyridinium ion. In the representation with aromatic bonds, the nitrogen atoms have the same connectivity, making it difficult to correctly assign formal charges.



A Current trail: N/A

B Current trail: $C_1=C_2–C_3$

C Current trail: $C_1=C_2–{*}C_3=C_4–C_5=N_6–{*}C_7=C_8–C_9=C_4$

D Current trail: $C_1=C_2–{*}C_3=C_4–C_5=N_6–C_7–C_8=C_9–C_4$

E Current trail: $C_1=C_2–{*}C_3=C_4–C_5=N_6–C_7–C_8=C_9–C_4$_$C_7=C_{11}–O_{12}$_$C_3–C_{10}$

F Current trail: $C_1=C_2–C_3–{*}C_4=C_5–N_6$

Current score: N/A
Best score: N/A

G Current trail: $C_1=C_2–C_3–{*}C_4=C_5–{*}N_6–{*}C_7=C_8–C_9=C_4$

Current score: N/A
Best score: N/A

H Current trail: $C_1=C_2–C_3–{*}C_4=C_5–{*}N_6–C_7–C_8=C_9–C_4$_$C_7=C_{11}–O_{12}$_$C_3=C_{10}$

Current score: 12
Best score: 12

I Current trail: $C_1=C_2–C_3–{*}C_4=C_5–N_6=C_7–C_8=C_9–C_4$_$C_7–C_{11}=O_{12}$_$C_3=C_{10}$

Current score: 11
Best score: 11

J Current trail: $C_1=C_2–C_3–C_4–C_5=N_6–{*}C_7=C_8–C_9=C_4$_$C_7–C_{11}=O_{12}$_$C_3=C_{10}$

Current score: 11
Best score: 11

Figure 4.

Step-by-step example of the workings of the “resolution of resonance” algorithm. Untraversed aromatic bonds, the order of which must be determined, are marked by dashed lines; short solid lines are single bonds to hydrogens or other radical groups and are ignored by the algorithm. Red and green arrows represent destructive and constructive backtracking, respectively. In the trail, points where an unexplored choice (of charge or double bond) exists are marked by an asterisk (*) and jumps following constructive backtracking are marked by an underscore (_). A full discussion of steps **A**–**H** can be found in the text.

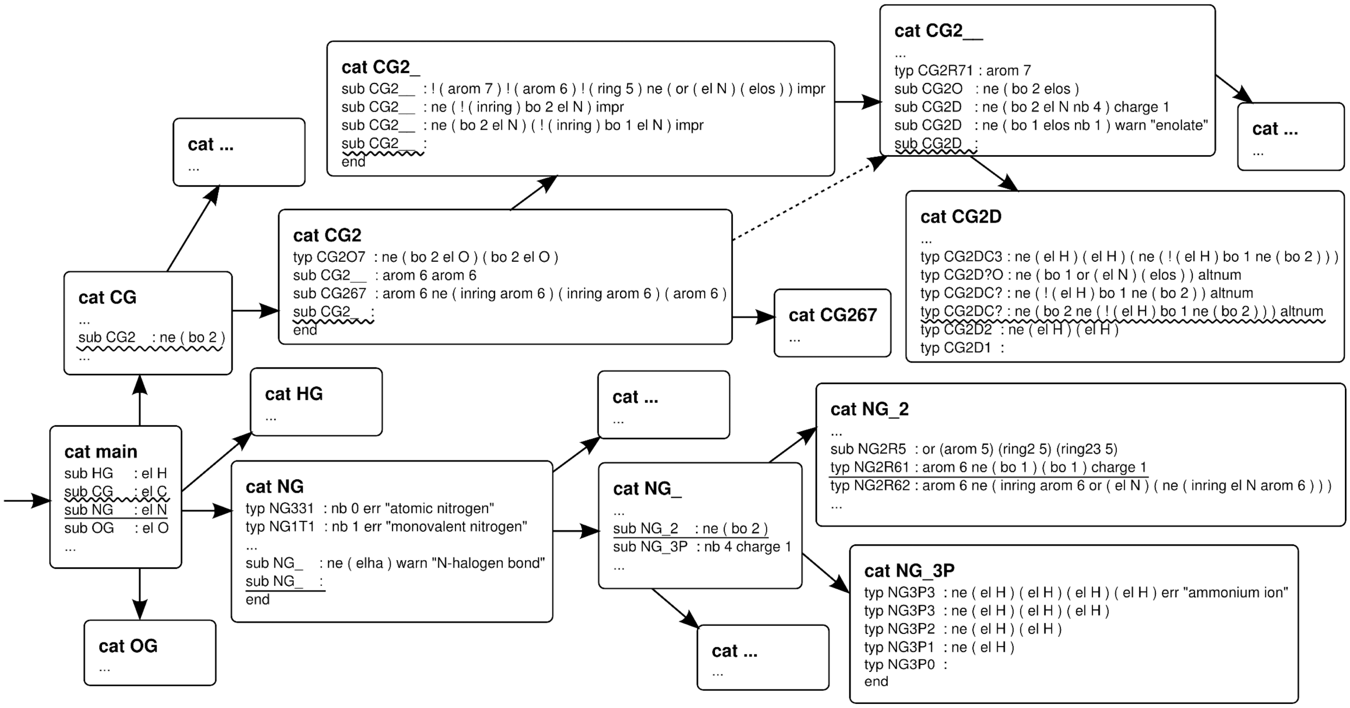

cat CG2_
sub CG2__ : ! (arom 7) ! (arom 6) ! (ring 5) ne (or (el N) (elos)) impr
sub CG2__ : ne (! (inring) bo 2 el N) impr
sub CG2__ : ne (bo 2 el N) (! (inring) bo 1 el N) impr
sub CG2__ :
end
cat CG2__
...
typ CG2R71 : arom 7
sub CG2O : ne (bo 2 elos)
sub CG2D : ne (bo 2 el N nb 4) charge 1
sub CG2D : ne (bo 1 elos nb 1) warn "enolate"
sub CG2D :
cat ...
...
cat ...
...
cat CG2
typ CG2O7 : ne (bo 2 el O) (bo 2 el O)
sub CG2__ : arom 6 arom 6
sub CG267 : arom 6 ne (inring arom 6) (inring arom 6) (arom 6)
sub CG2_ :
end
cat CG2D
...
typ CG2DC3 : ne (el H) (el H) (ne (! (el H) bo 1 ne (bo 2)))
typ CG2D?O : ne (bo 1 or (el N) (elos)) altnum
typ CG2DC? : ne (! (el H) bo 1 ne (bo 2)) altnum
typ CG2DC? : ne (bo 2 ne (! (el H) bo 1 ne (bo 2))) altnum
typ CG2D2 : ne (el H) (el H)
typ CG2D1 :
cat CG
...
sub CG2 : ne (bo 2)
...
cat CG267
...
cat HG
...
cat ...
...
cat NG_2
...
sub NG2R5 : or (arom 5) (ring2 5) (ring23 5)
typ NG2R61 : arom 6 ne (bo 1) (bo 1) charge 1
typ NG2R62 : arom 6 ne (inring arom 6 or (el N) (ne (inring el N arom 6)))
...
cat main
sub HG : el H
sub CG : el C
sub NG : el N
sub OG : el O
...
cat NG
typ NG331 : nb 0 err "atomic nitrogen"
typ NG1T1 : nb 1 err "monovalent nitrogen"
...
sub NG_ : ne (elha) warn "N-halogen bond"
sub NG_ :
end
cat NG_
...
sub NG_2 : ne (bo 2)
sub NG_3P : nb 4 charge 1
...
cat NG_3P
typ NG3P3 : ne (el H) (el H) (el H) (el H) err "ammonium ion"
typ NG3P3 : ne (el H) (el H) (el H)
typ NG3P2 : ne (el H) (el H)
typ NG3P1 : ne (el H)
typ NG3P0 :
end
cat OG
...
cat ...
...

Figure 5.

Decision tree representation of the parts of the rule file that are discussed in the case studies. Specifically, the rules that are triggered in the pyrrolidinium N (Figure 3) and the 1,3-dipentene C4 (Figure 6) case studies are respectively underlined with straight and wavy lines.

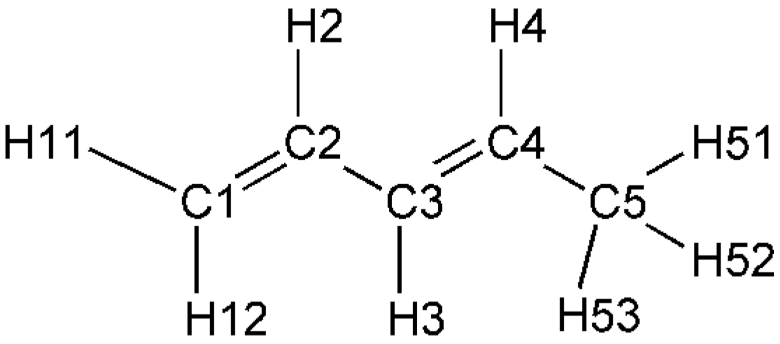


Figure 6.

1,3-dipentene, the subject of the second case study.

Table 1

Overview of all the *condition* keywords supported by the atom typer. The type of condition can be “atom”: evaluate an atomic property, “topology”: related to the connectivity of the molecular graph, “bond”: evaluate a bond property, “logic”: logical operator or “action”: *optional action* (see text). Note that the latter are not conditions in the sense that they don’t test anything - performing an action instead - although they could be regarded as conditions that always evaluate true.

Condition	**Meaning**	**Type**
el <element>	element	atom
elha	halogen	atom
elos	chalcogen	atom
nb <#>	valence	atom
rings <#>	# of rings	atom
ring3 <size>	a*ll-sp*^{3} ring	atom
ring2 <size>	a*ll-sp*^{2} ring	atom
arom <size>	*aromatic* ring	atom
ring23 <size>	*mixed sp*^{2}*/sp*^{3} ring	atom
ring <size>	any ring	atom
self	parent atom	atom/topology
ne (<cond>) (<cond>) …	neighbor	topology
bo <order>	bond order	bond
inring	bond in ring	bond
! (<cond>)	not operator	logic
or (<cond>) (<cond>) …	or operator	logic
charge <#>	formal charge	action
impr	improper dihedral	action
warn “<message>”	(non-fatal) warning	action
err “<message>”	(fatal) error	action
altnum	alternate numbering	action