RDChiral: An RDKit Wrapper for Handling Stereochemistry in Retrosynthetic Template Extraction and Application

As Published: 10.1021/acs.jcim.9b00286

Publisher: American Chemical Society (ACS)

Persistent URL: https://hdl.handle.net/1721.1/134762

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

RDChiral: An RDKit Wrapper for Handling Stereochemistry in Retrosynthetic Template Extraction and Application

Connor W. Coley, William H. Green,* and Klavs F. Jensen*

Department of Chemical Engineering, MIT, Cambridge, MA

E-mail: whgreen@mit.edu; kfjensen@mit.edu

Abstract

There is a renewed interest in computer-aided synthesis planning, where the vast majority of approaches require the application of retrosynthetic reaction templates. Here, we introduce an open source Python wrapper for RDKit designed to provide consistent handling of stereochemical information in applying retrosynthetic transformations encoded as SMARTS strings. RDChiral is designed to enforce the introduction, destruction, retention, and inversion of tetrahedral centers as well as the cis/trans chirality of double bonds. We also introduce an open source implementation of a retrosynthetic template extraction algorithm to generate SMARTS patterns from atom-mapped reaction SMILES strings. In this manuscript, we describe the implementation of these two pieces of code and illustrate their use through many examples.

Introduction

The rising availability of reaction corpora, hardware for rapid computing, and algorithms for efficient search have led to a renewed interest in computer-aided synthesis planning

(CASP).[1–5] The majority of CASP programs are based around the use of reaction templates–subgraph patterns that describe the changes in connectivity between a product molecule and its corresponding reactant(s)–to generate recommendations for retrosynthetic disconnections.

The earliest CASP programs sought to directly codify expert chemist knowledge about what reactions are allowed.[6] This "expert approach" is reflected by Synthia (formerly Chematica[4]), which now contains around 70,000 hand-encoded reaction transformation rules and has been successfully used to plan synthetic routes to a number of complex products.[4] Significantly smaller rule sets have been made public and popularized, like Hartenfeller et al.'s set of 58 reactions popular in medicinal chemistry,[7] while others are integrated into closed-source commercial programs.[8,9]

As an alternative to manually-encoding allowable transformations, there have been heuristics developed for algorithmic extraction.[10–20] These algorithms build generalized rules from known reaction examples. Broadly speaking, they all identify the atoms that change connectivity as the reaction center. Different levels of generalization can then be used to extend that reaction center to include varying numbers of neighbors, either using a fixed distance or using heuristics that decide which neighboring atoms are *relevant*.

There are additional approaches to computer-aided retrosynthesis that avoid the need for reaction templates entirely. These include sequence-to-sequence models,[21] similarity-based methods,[22] and some graph-based methods that have thus far been applied to the problem of forward prediction.[23]

In this manuscript, we describe our approach to retrosynthetic template extraction and application. With the exception of a recently open sourced implementation in C++ from Watson et al.,[20] there have been no published algorithms for template extraction to enable complete reproducibility. Moreover, this implementation does not explicitly discuss stereochemistry; stereochemical handling is rarely mentioned in the context of SMARTS template application, and there is no detailed description of doing so using open source tools.[4] The specific contributions of this work are two-fold:

(1) An open source Python implementation of retrosynthetic template *extraction*, designed to operate on atom-mapped SMILES[24] strings (*i.e.*, where the correspondence between atoms in the product and atoms in the reactants are known) and generate generalized SMARTS[25] patterns. While the procedure we describe also works for reactions in the forward synthetic direction, we focus on the retrosynthetic direction where templates are applied to product molecules to generate one or more reactant/precursor molecules.

(2) RDChiral, an open source Python implementation of retrosynthetic template *application*, designed to provide consistent handling of stereochemistry defined by template SMARTS strings.

Implementation

Both software contributions make extensive use of RDKit (version 2018.09.1),[26] which has become one of the most widely-used frameworks for cheminformatics research.

Template extraction procedure

The pipeline for template extraction begins with atom-mapped SMILES strings in the form `reactants>>major_product`. Only molecules that contribute heavy atoms (non-hydrogen) to the major product are required; spectator molecules are discarded. Not all product atoms need to be atom mapped; it is assumed that small fragments of fewer than 5 atoms (e.g., a halogen, oxygen) may come from unlisted reagents (e.g., Br_2, O_2) that were mistakenly excluded from the dataset. Reactant molecules that are only partially mapped are allowed, provided that the unmapped atoms correspond to leaving groups. Each atom map number can only appear once in the product.

1. **Check for parsing errors.** If any of the reactant or product atoms fail to be parsed and sanitized by RDKit, skip. This is uncommon but can result from perceived valence rule

violations if the source SMILES strings were prepared by another program. Molecules are sanitized, which includes perceiving aromaticity and converting structures to their non-Kukulé form.

2. **Check for unmapped product atoms.** If the number of unmapped product atoms exceeds a maximum allowed threshold (five), skip. Otherwise, record the fragment that must be contributed by an unreported reagent. For example, the reaction `c1ccccc1>>[Br]c1ccccc1` (mapping omitted for brevity) suggests that a bromine atom, `[Br]`, must come from a reagent.

3. **Determine which atoms, by map number, have changed.** First, identify the mapping numbers of the atoms in the reactants and product. For each atom in the product, determine if that atom's local properties are identical to how they were in the reactants. Two atoms are considered identical if they have the same SMARTS pattern, atomic number, total number of hydrogens, formal charge, degree, number of radical electrons, aromaticity, and bond order and atomic number of neighboring atoms. Any reactant atoms that are mapped but do not appear in the product are included in the list of changed atoms as well.

   A more detailed version of this analysis is performed for any atoms that are tetrahedral centers that could have specified chirality: quartenary carbon with nonidentical side chains. The chirality of the atom–as determined by its clockwise/counterclockwise orientation and neighboring atoms–is checked. Note that this must be a *local* check, as a chiral center's absolute stereochemical assignment (R/S) can change as a result of changes to the Cahn Ingold Prelog (CIP)[27] priority of its side chains (Figure 1); the same holds true for absolute cis/trans (Z/E) assignment.

4. **Define a SMARTS pattern to describe the reactants.**

   a. **Define strict SMARTS for each reacting atom.** For each atom that changed between the reactants and products, including unmapped atoms belonging to leaving

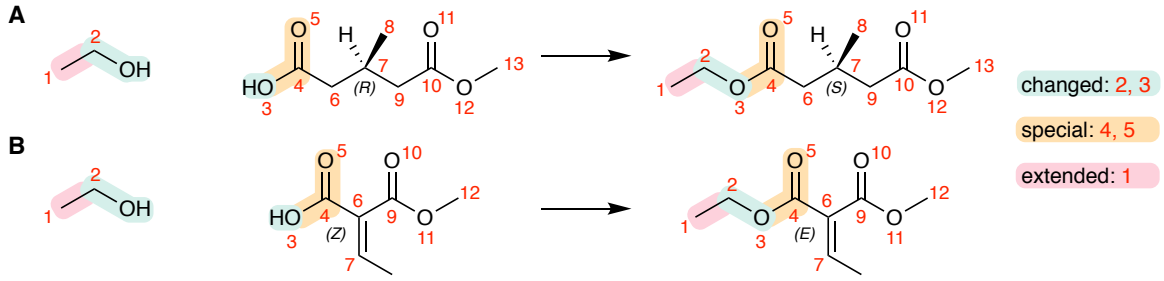

Figure 1: Reactions where there is a change in absolute stereochemical assignment, but the (A) atom or (B) bonds that are affected are not part of the reaction center and thus do not undergo a change in their *local* configuration.

groups, define a strict SMARTS pattern. This SMARTS pattern includes atomic number, aromaticity, tetrahedral chirality (if applicable), number of hydrogens, degree, formal charge, and atom map number.

b. **Identify atom membership in special functional groups.** When reacting atoms are part of specific substructural motifs, that whole motif should be included in the template. As an example, if a reacting atom is adjacent to an alkene or carbonyl, then the presence of that alkene or carbonyl is likely important for its reactivity. The two additional atoms belonging to the alkene or carbonyl are added to the list of neighboring atoms.

   We have defined roughly 30 such groups, including carboxylic acids, amides/sulfamides, boronic acids/esters, common protecting groups, alkenes/imines, alkynes/nitriles, adjacencey to alkenes/alkynes/carbonyls, organometallics, diazo groups, adjacency to a heteroatom in a ring, two atoms away from a heteroatom in an aromatic ring, trifluoromethyls. If reacting atoms belong to alkenes where cis/trans chirality is specified, the neighboring atoms required to fully define that local configuration–irrespective of the priority of side chains–are included as well.

c. **Define generalized SMARTS for each neighboring atom.** For every atom that is a neighbor of a reacting atom (or belongs to a special functional group), define a SMARTS atom pattern. For terminal atoms with degree one, this SMARTS pat-

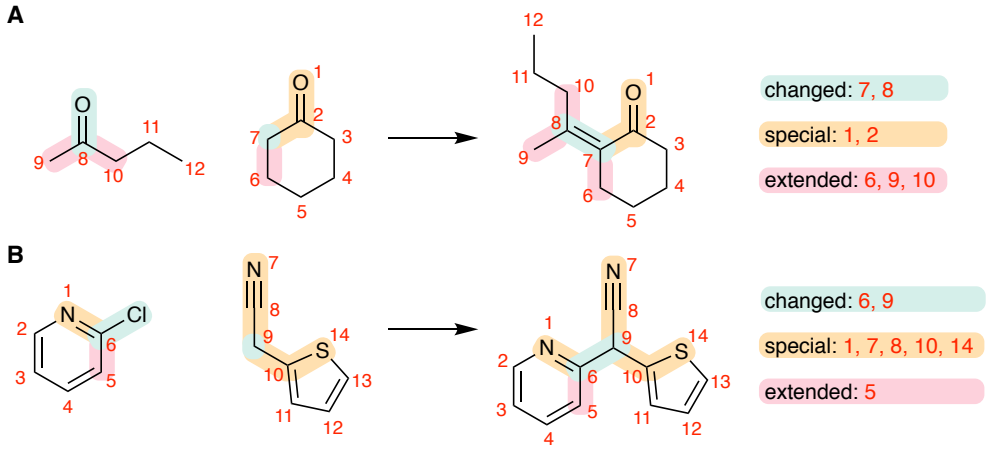

Figure 2: Two example reactions where atoms belonging to "special" functional groups are added to the reaction template. This helps ensure that the relevant chemical context for the reaction is present, even if more than a single bond away from the reacting atoms.

tern includes its atomic number, degree, number of hydrogens, degree, formal charge, and atom map number (if applicable). For atoms with degree greater than one, this SMARTS pattern includes only its atomic number, aromaticity, and formal charge.

d. **Generate an overall SMARTS pattern for the reactants.** Using RDKit's `MolFragmentToSmiles` with atom SMILES tokens replaced by their custom SMARTS pattern, generate an overall SMARTS pattern for the reactants. All hydrogens and bonds are included explicitly.

   The generated SMARTS is matched back onto the reactant molecules to ensure a match. If there is a mismatch due to tetrahedral chirality, the orientation of chiral centers is flipped until a match can be found.

e. **Record auxiliary information about the transformation.** If only one reactant molecule has changed, record that the eventual SMARTS transformation should only be applied as an intramolecular reaction. If two reactant molecules have changed, but they have identical SMILES strings when not atom mapped, record that the SMARTS transformation should only be applied as a dimerization reaction.

5. **Define a SMARTS pattern to describe the product.**

a. **Define strict SMARTS for each reacting or unmapped atom.** For each atom that changed between the reactants and products, define a strict SMARTS pattern. This SMARTS pattern includes atomic number, aromaticity, tetrahedral chirality (if applicable), number of hydrogens, degree, formal charge, and atom map number. Also include any unmapped product atoms with this strict definition.

b. **Include additional atoms corresponding to non-reacting reactant atoms.** Identify mapped atoms in the reactants that were added to the SMARTS pattern but did not react (i.e., mapped atoms neighboring the reacting atoms or added as part of special functional groups). For each, generate a generalized SMARTS. For terminal atoms with degree one, this SMARTS pattern includes its atomic number, degree, number of hydrogens, degree, formal charge, and atom map number (if applicable). For atoms with degree greater than one, this SMARTS pattern includes only its atomic number, aromaticity, and formal charge.

c. **Generate an overall SMARTS pattern for the product.** Using RDKit's `MolFragmentToSmiles` with atom SMILES tokens replaced by their custom SMARTS pattern, generate an overall SMARTS pattern for the product. All hydrogens and bonds are included explicitly.

   The generated SMARTS is matched back onto the product molecules to ensure a match. If there is a mismatch due to tetrahedral chirality, the orientation of chiral centers is flipped until a match can be found.

6. **Merge the two patterns into an overall retrosynthetic reaction SMARTS.** Attempt to canonicalize the order of disconnected reactant/product fragments by sorting the SMARTS patterns alphabetically. Reassign atom map numbers in the string to replace the original map numbers from the reactant product molecules with a continuous sequence starting at 1. Create the final retrosynthetic template as a concatenation `product_smarts>>reactants_smarts`.

Template application procedure

Template application is based around RDKit's `RunReactants`. The initial set of outcomes is generated by applying the template SMARTS string to an *achiral* version of the product structure. What follows are a number of steps to ensure that (a) a match should have occurred, and (b) the chirality of the resulting precursors is as intended. To identify the atom-to-atom correspondence between the input product and generated reactants, we assign unique dummy isotope values to each product atom; isotope values of reactants can then be used as the atom mapping.

1. **Initialize rdchiralReaction object from SMARTS.** Initialize a reaction from the SMARTS pattern. Also create auxiliary lists/dictionaries for faster processing: a dictionary mapping the atom map number in a template to the atom ID, a list of atoms in the template that could have tetrahedral chirality specified, a list of double bonds that could have cis/trans chirality specified.

2. **Initialize rdchiralReactants object from SMILES.** Initialize a molecule from the input SMILES. Create a copy of that molecule without stereochemical information. Identify all tetrahedral centers and double bonds and whether they have their chirality specified (or whether that is even possible, due to symmetry). Create auxiliary lists/dictionaries for faster processing, including lists of these directional double bonds and tetrahedral centers.

3. **Generate outcomes using the achiral input compound.** The initial set of precursors are generated using RDKit's `RunReactants` with an achiral copy of the input molecule. This will lead to *at least as many* matches as we would like to consider valid.

4. **Ensure that tetrahedral center chirality matches between the input molecule and the template.** If a reaction template has tetrahedral chirality defined in its SMARTS pattern, then the input molecule must also have its chirality defined. When

there is only one tetrahedral center, it must be well-defined but need not match. When there are multiple tetrahedral centers, they must all match the template exactly or all be mirror images. That is, the same template will match two compounds that are enantiomers but will distinguish between diastereomers. This check is again done *locally*, as absolute stereocenter assignment can change due to transformations far from the stereocenter.

5. **Ensure that double bond chirality matches between the input molecule and the template.** For each alkene in the input molecule, check if all of the atoms that would be required to define cis/trans chirality were included in the reaction template. If the reaction template does not have defined chirality but the product molecule does, do not allow a match. If the reaction template does have defined chirality, ensure that the product molecule matches. This check is slightly complicated by the fact that, as with checking tetrahedral centers, it must be done based on local connectivity and not the absolute cis/trans assignment. The implied stereochemsitry of double bonds found in aliphatic rings are taken into account during this consistency check.

6. **Ensure intramolecular reactions applied properly.** Another quirk of retrosynthetic template application is that intramolecular ring opening reactions can sometimes lead to accidental fragmentation. Duplicate atoms in the reactants are detected and the two precursor fragments are re-recombined.

7. **Check the chirality of tetrahedral precursor atoms.** If a precursor atom was generated by the reaction SMARTS and not copied over (i.e., part of a leaving group), it will have the correct chirality. If the atom did not match any part of the template, then the chirality is directly copied from the reactants. If the atom matched part of the template, then we must check whether it was possible for that template to have specified the chirality: if the atom in the template could *not* have had its chirality specified, then the chirality of the precursor atom is copied from the input molecule.

There are several cases to consider if the template input atom could have had its chirality specified. If the corresponding template output atom has unspecified chirality, then the generated precursor atom should have its chirality stripped (for a retrosynthetic template, this means that this is a stereoselective reaction where the selectively is attributable to a reagent or catalyst, e.g., a proline-catalyzed aldol reaction). If the template input atom is unspecified, then the generated precursor atom has its chirality copied from the template output atom. If both the template input and template output atoms have specified chirality, then we must check whether the template describes the *retention* of chirality or *inversion*, again using a local definition. The chirality of the generated precursor atom is copied from the input atom and retained or inverted according to the template.

8. **Check the chirality of precursor double bonds.** If both atoms across a carbon-carbon or carbon-nitrogen double bond in the generated precursor matched the template and it was possible for that template to specify cis/trans chirality, then they will already have the correct chirality. If both atoms were created by the template (retrosynthetically, were part of a chiral leaving group) then they will also have been instantiated with the correct chirality. Otherwise, as in the tetrahedral case, the chirality of the generated precursor double bond should be copied from the input molecule. This is again somewhat complex, as there are many ways to locally define the orientation of a double bond in the SMILES language and the absolute assignment of a double bond cannot be used to check for consistency (see Figure 3).

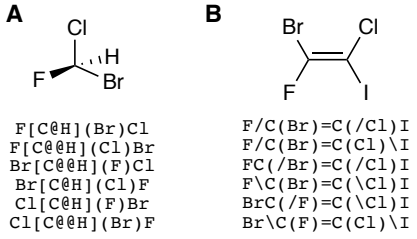

Figure 3: Nonexhaustive set of SMILES strings that all refer to the same molecular structure, for (A) a molecule with tetrahedral chirality, and (B) a molecule with cis/trans chirlaity.

9. **Merge enantiomeric precursors into a single racemic precursor.** In cases where the set of possible precursors includes two enantiomers, we can optionally merge these into a single racemic precursor. This can happen in cases like the elimination of a chiral alcohol to form a racemic alkene; though the retrosynthetic template will generate chiral precursors from the alkene, the elimination product of either enantiomer will be the same alkene and so it is appropriate to combine the precursor suggestions into the racemic alcohol.

For speed, when the same template or same product molecule will be used numerous times (e.g., when applying a full library of retrosynthetic templates to a library of candidate product molecules), a custom initialization can precompute many required properties to reduce computational cost.

Results

Reaction data source

All examples in Figure 4 and Figure S1 are from reactions contained in the open source USPTO reaction dataset containing ca. 1.8M reactions.[28] This dataset has been previously mapped using the Indigo toolkit so that all heavy atoms in the products are mapped.

Template extraction

Several example reactions are shown in Figure 4 that show the identification of changed atoms (cyan), adjacent atoms belonging to special functional groups (dark yellow), and then the final extension to any atoms neighboring the changed atoms that were not part of such groups (pink). The resulting templates have varying degrees of specificity.

Figure 4A defines a highly-general aromatic bromination with *N*-Bromosuccinimide (NBS). Figure 4B defines the double-alkylation of a catechol derivative with 1,2-dibromoethane. Fig-

ure 4C defines the acid hydrolysis of an alkyl nitrile using sulfuric acid. Figure 4D defines an amidation reaction between an aniline and an aliphatic acid chloride. Figure 4E produces a three-component reaction template between a primary thioamide, formaldehyde, and a disubstituted amine. Figure 4F is a simple deacetylation of a phenylacetate. Figure 4G defines a template to prepare a 4-chloropyrimidine from a tautomer of 4-hydroxypyrimidine. Figure 4H defines the alkylation of a 2-hydroxypyridine using a tosyl alkylate. Figure 4I defines an epoxide opening using a monosubstituted amine. Figure 4J is a simple ethyl ester hydrolysis. Figure 4K is a Knoevenagel condensation between a benzaldehyde and a cyanoacetate. Figure 4L is another bromination with NBS, but of a methyl group on an aromatic ring. Figure 4M is an organometallic reaction between an aryl lithium and a benzaldehyde.

Additional example reactions are shown in Figure S1 that show how the same process applies to reactions involving change in stereochemistry.

Template application

Several examples of retrosynthetic template application are shown in Figure 5 and Figure 6. These are designed to showcase a range of situations where we may or may not require different behavior than what RDKit's standard `RunReactants` can provide. The SMARTS string for each retrosynthetic template is included, as is a short description of the forward reaction. For all of these cases, RDChiral yields the "correct" precursors or lack thereof. The comparisons we show are meant to highlight the role of the RDChiral wrapper that is specifically tailored to retrosynthesis; they are not meant to reflect negatively on RDKit, which makes fewer assumptions about the intended use cases than we do.

Figure 5A is a single example of consistency between RDKit and RDChiral for the case where neither the template nor the input product molecule contain any stereochemical information. Because RDChiral is a wrapper for RDKit, there will almost always be agreement in this achiral case. Figure 5B shows the one exception where applying a retrosynthetic disconnection to cleave an ester can result in inadvertent fragmentation of a molecule, which

RDChiral can identify and correct for.

When the product molecule has stereochemical information that is not part of where the template is matched, the standard behavior of RDKit is what we would like: local configurations are copied to the generated precursors (see the agreement in Figure 5C). When the stereochemical information is partially in the template, there are certain cases (Figure 5D) where RDChiral must restore that information in the generated precursors.

Figure 5E demonstrates the most important role of RDChiral: preventing matches of chiral products with achiral templates when the substructure that matches the template could have specify chirality (i.e., the matching substructure fully contains the atoms and bonds required to define the chiral center). In these situations, generating precursors would mistakenly suggest that the reaction is guaranteed to be stereoselective or stereoretentive, which is *not* implied by the template. For both of the cases shown (defined tetrahedral center and defined double bond directionality), no precursors should be generated, and RDChiral ensures that this is the case. The reverse is also true (Figure 5F); matches are not allowed between chiral templates and achiral products.

The examples in Figure 6 show several cases where checking for consistency in retrosynthetic template application is important. Of particular note are the following: (1) double bonds found in aliphatic rings are implicitly cis, and will match templates requiring this double bond chirality (Figure 6A). And (2) inversion and retention of tetrahedral centers, as defined by the template, result in precursors whose chirality is consistent with or opposite the chirality in the product (rather than being generated based on the right side of the template SMARTS) (Figure 6CD). In the cases where each side of the template contains a single stereocenter, the specific clockwise/counterclockwise specification is ignored; the information we parse from the reaction SMARTS is simply whether that stereocenter is inverted or preserved, as evidenced by the multiple SMARTS strings resulting in equivalent transformations. However, when multiple tetrahedral centers are present, the diastereomerism must match (Figure 6G).

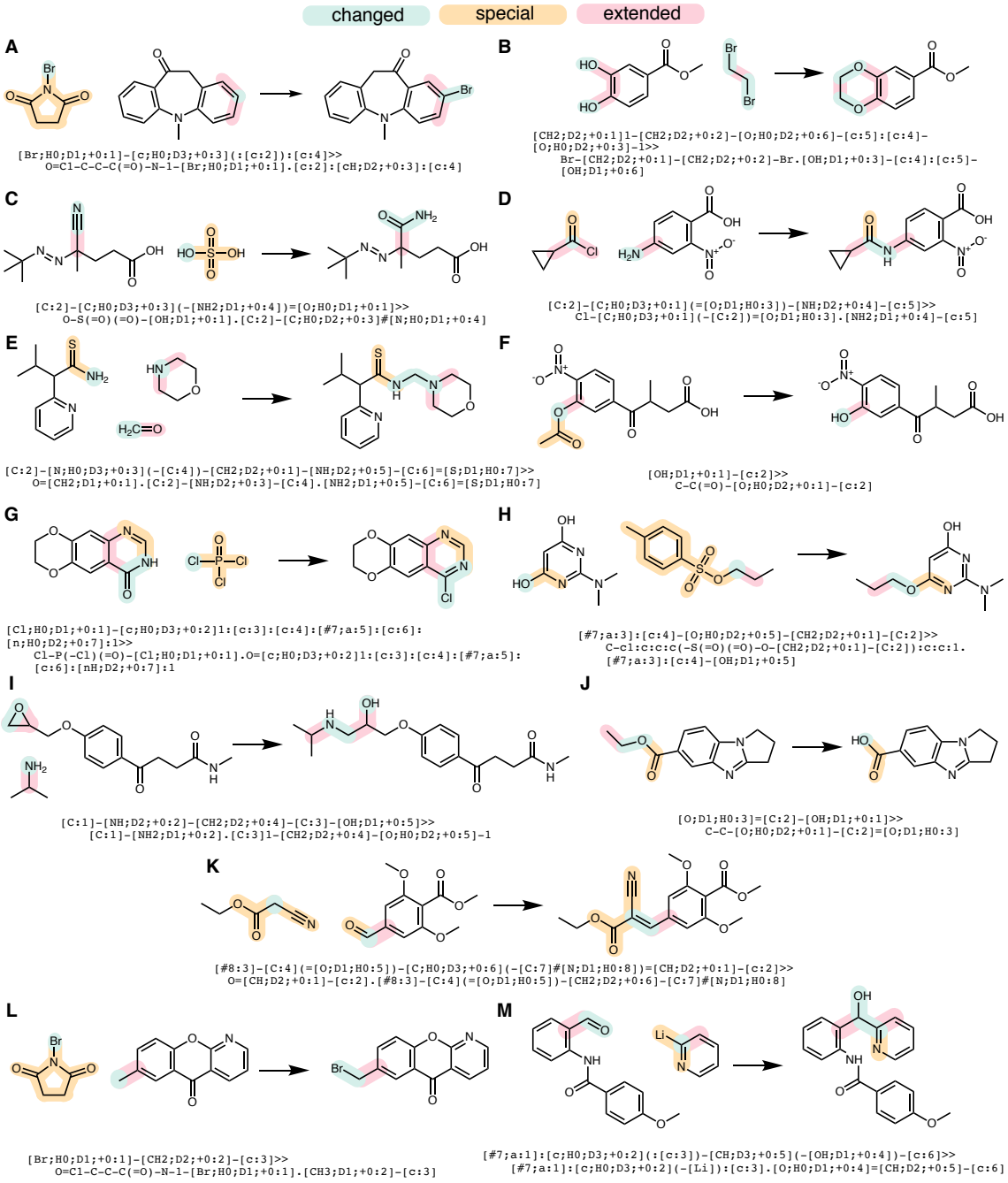

Figure 4: Examples of reactions from the USPTO and the retrosynthetic templates we extract from them. Atom mapping is omitted for brevity, but is unambiguous in all cases shown. Spectator molecules are also omitted, as they do not contribute heavy atoms to the product and are not included in the resulting reaction template.

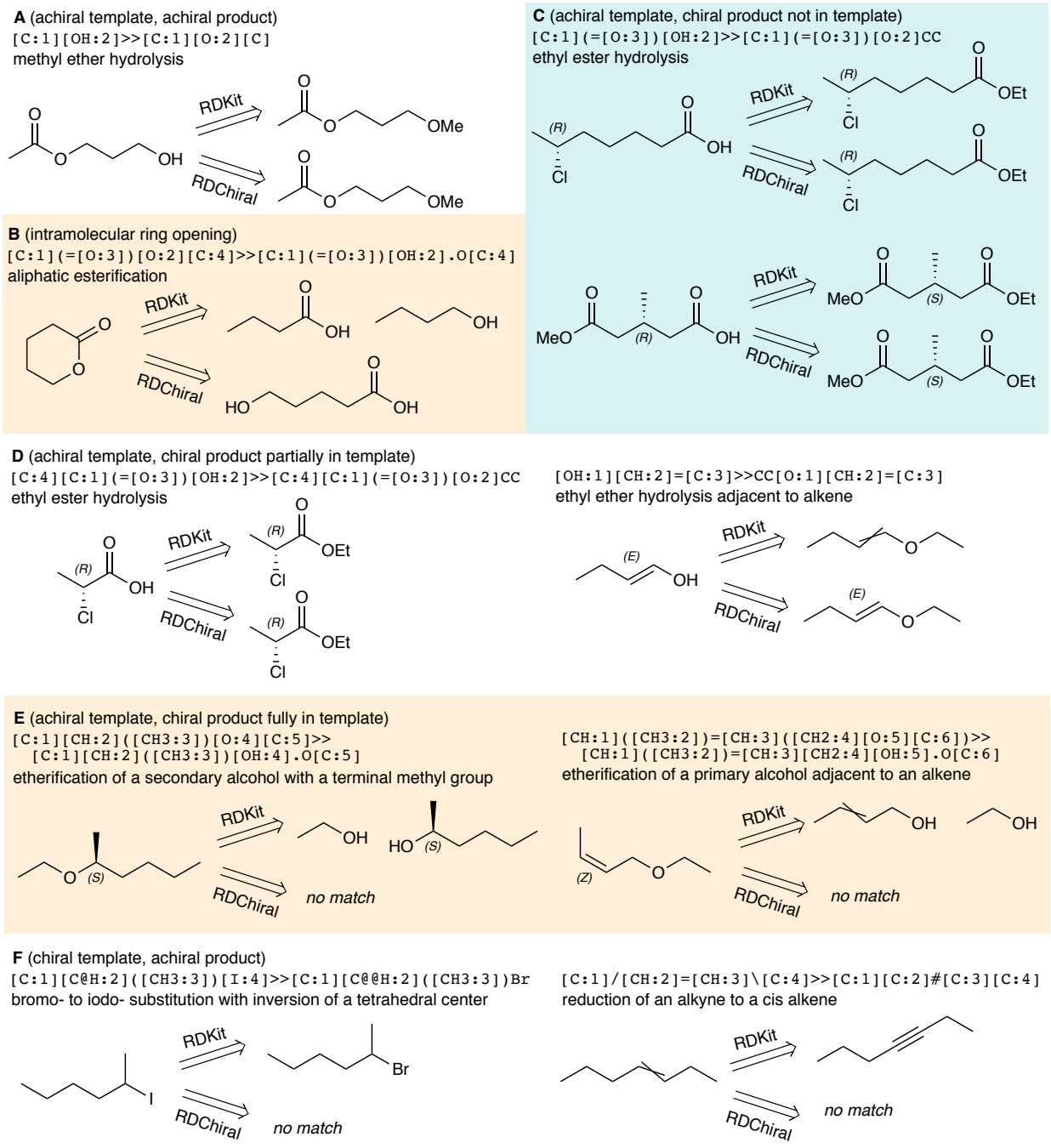

Figure 5: Examples of difference in retrosynthetic template application between the standard RDKit implementation and RDChiral for a variety of cases. (A) achiral template matched to an achiral product; (B) achiral template matched to an achiral product resulting in an intramolecular ring opening; (C) achiral template applied to a chiral product, where the chiral center is not part of the template; (D) achiral template applied to a chiral product, where the chiral center is part of the template but not fully specified; (E) achiral template applied to a chiral product, where the chirality is fully within the template; (F) chiral template applied to an achiral product.

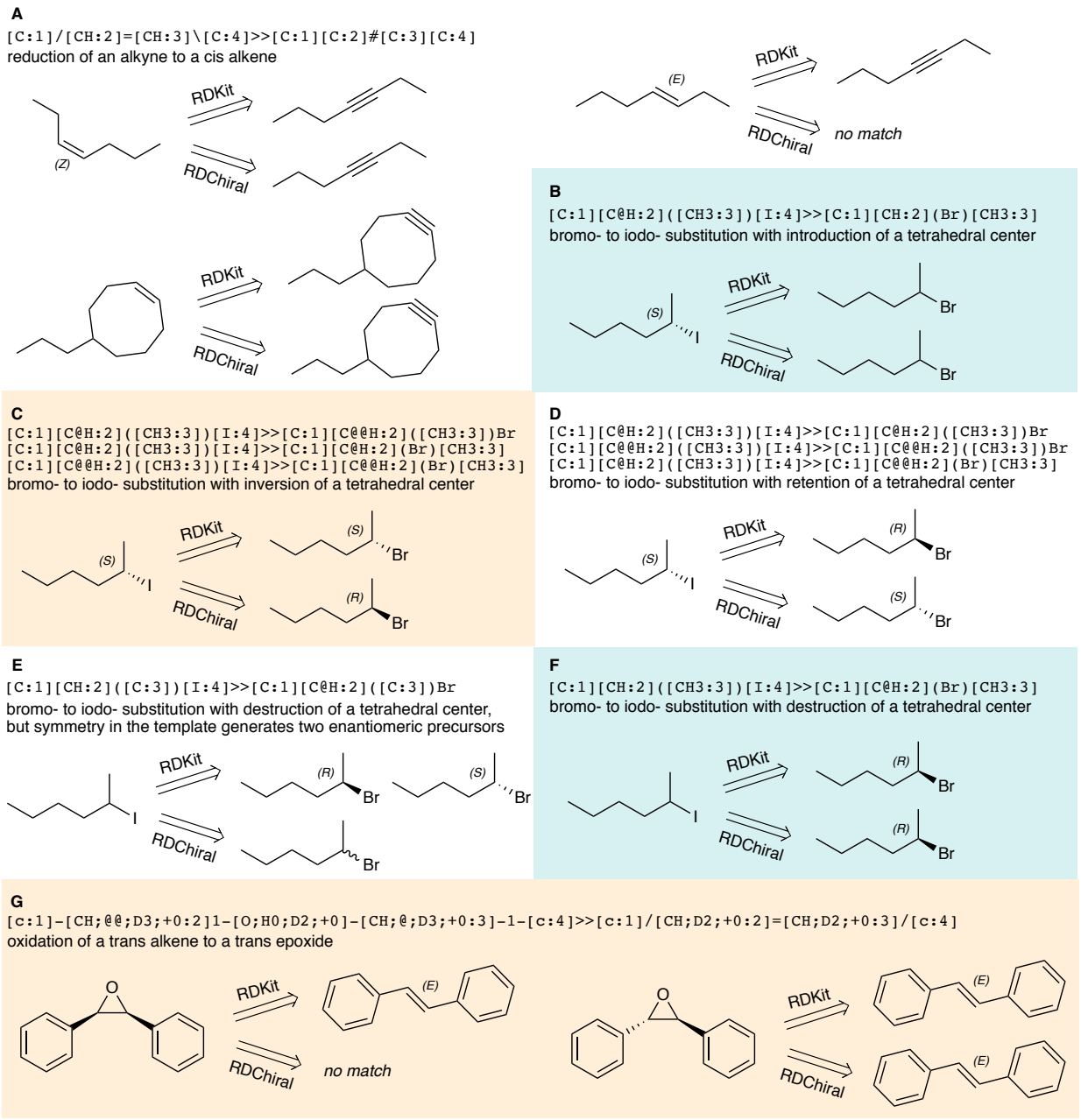

Figure 6: Additional examples of differences in retrosynthetic template application between the standard RDKit implementation and RDChiral for a variety of cases. (A) chiral template to produce a cis alkene requires a cis product, but recognizes implicit definitions in ring structures; (B) template that removes a tetrahedral center from the product; (C) template to invert a tetrahedral center, with several equivalent SMARTS strings; (D) template to retain a tetrahedral center, with several equivalent SMARTS strings; (E) template describing the destruction of a chiral center yields racemic precursors due to symmetry on the right side of the template; (F) template describing the destruction of a chiral center yields single enantiomer as specified by right side of the template; (G) template describing diastereoselective epoxidation discriminates between diastereometers but not enantiomers.

Conclusion

We have described two interrelated pieces of open source Python software, based on RDKit, that (A) extract retrosynthetic templates from atom-mapped SMILES strings using a dynamic definition of the relevant context surrounding the reaction center, and (B) apply retrosynthetic templates in a manner that is faithful to the definition of stereochemical information (or lack thereof) in the template SMARTS or input product molecule. This software has proved essential for our own synthetic planning workflows[29] in enabling increasingly complex small molecule targets that necessitate careful consideration of chirality.

Acknowledgement

This work was supported by the DARPA Make-It program under contract ARO W911NF-16-2-0023. C.W.C. received additional funding from the NSF GRFP under Grant No. 1122374. We would like to thank Mike Fortunato and Thomas J. Struble for helpful comments on this manuscript and with aspects of the code.

Supporting Information Available

All code can be found at `https://github.com/connorcoley/rdchiral` in addition to Jupyter notebooks containing all of the examples included in this manuscript. Additional figures can be found in the supporting information.

References

(1) Todd, M. H. Computer-Aided Organic Synthesis. *Chem. Soc. Rev.* **2005**, *34*, 247–266.

(2) Cook, A.; Johnson, A. P.; Law, J.; Mirzazadeh, M.; Ravitz, O.; Simon, A. Computer-

Aided Synthesis Design: 40 Years On. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2012**, *2*, 79–107.

(3) Warr, W. A. A Short Review of Chemical Reaction Database Systems, Computer-Aided Synthesis Design, Reaction Prediction and Synthetic Feasibility. *Mol. Inform.* **2014**, *33*, 469–476.

(4) Szymkuc, S.; Gajewska, E. P.; Klucznik, T.; Molga, K.; Dittwald, P.; Startek, M.; Bajczyk, M.; Grzybowski, B. A. Computer-Assisted Synthetic Planning: The End of the Beginning. *Angew. Chem. Int. Ed.* **2016**, *55*, 5904–5937.

(5) Coley, C. W.; Green, W. H.; Jensen, K. F. Machine Learning in Computer-Aided Synthesis Planning. *Acc. Chem. Res.* **2018**, *51*, 1281–1289.

(6) Corey, E. J.; Jorgensen, W. L. Computer-Assisted Synthetic Analysis. Synthetic Strategies Based on Appendages and the Use of Reconnective Transforms. *J. Am. Chem. Soc.* **1976**, *98*, 189–203.

(7) Hartenfeller, M.; Eberle, M.; Meier, P.; Nieto-Oberhuber, C.; Altmann, K.-H.; Schneider, G.; Jacoby, E.; Renner, S. A Collection of Robust Organic Synthesis Reactions for in Silico Molecule Design. *J. Chem. Inf. Model.* **2011**, *51*, 3093–3098.

(8) Konze, K.; Bos, P.; Dahlgren, M.; Leswing, K.; Tubert-Brohman, I.; Bortolato, A.; Robbason, B.; Abel, R.; Bhat, S. Reaction-based Enumeration, Active Learning, and Free Energy Calculations to Rapidly Explore Synthetically Tractable Chemical Space and Optimize Potency of Cyclin Dependent Kinase 2 Inhibitors. `https://chemrxiv.org/articles/Reaction-based_Enumeration_Active_Learning_and_Free_Energy_Calculations_to_Rapidly_Explore_Synthetically_Tractable_Chemical_Space_and_Optimize_Potency_of_Cyclin_Dependent_Kinase_2_Inhibitors/7841270`.

(9) ChemAxon Reactor. `https://chemaxon.com/products/reactor`.

(10) Gelernter, H.; Rose, J. R.; Chen, C. Building and Refining a Knowledge Base for Synthetic Organic Chemistry Via the Methodology of Inductive and Deductive Machine Learning. *J. Chem. Inf. Comput. Sci.* **1990**, *30*, 492–504.

(11) Satoh, H.; Funatsu, K. SOPHIA, a Knowledge Base-guided Reaction Prediction System-utilization of a Knowledge Base Derived From a Reaction Database. *J. Chem. Inf. Comput. Sci.* **1995**, *35*, 34–44.

(12) Satoh, K.; Funatsu, K. A Novel Approach to Retrosynthetic Analysis Using Knowledge Bases Derived From Reaction Databases. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 316–325.

(13) Law, J.; Zsoldos, Z.; Simon, A.; Reid, D.; Liu, Y.; Khew, S. Y.; Johnson, A. P.; Major, S.; Wade, R. A.; Ando, H. Y. Route Designer: A Retrosynthetic Analysis Tool Utilizing Automated Retrosynthetic Rule Generation. *J. Chem. Inf. Model.* **2009**, *49*, 593–602.

(14) Christ, C. D.; Zentgraf, M.; Kriegl, J. M. Mining Electronic Laboratory Notebooks: Analysis, Retrosynthesis, and Reaction Based Enumeration. *J. Chem. Inf. Model.* **2012**, *52*, 1745–1756.

(15) Bogevig, A.; Federsel, H.-J.; Huerta, F.; Hutchings, M. G.; Kraut, H.; Langer, T.; Low, P.; Oppawsky, C.; Rein, T.; Saller, H. Route Design in the 21st Century: The ICSYNTH Software Tool as an Idea Generator for Synthesis Prediction. *Org. Process Res. Dev.* **2015**, *19*, 357–368.

(16) Coley, C. W.; Barzilay, R.; Jaakkola, T. S.; Green, W. H.; Jensen, K. F. Prediction of Organic Reaction Outcomes Using Machine Learning. *ACS Cent. Sci.* **2017**, *3*, 434–443.

(17) Segler, M. H. S.; Waller, M. P. Neural-Symbolic Machine Learning for Retrosynthesis and Reaction Prediction. *Chem. Eur. J.* **2017**, *23*, 5966–5971.

(18) Segler, M. H. S.; Preuss, M.; Waller, M. P. Planning Chemical Syntheses With Deep Neural Networks and Symbolic AI. *Nature* **2018**, *555*, 604–610.

(19) Baylon, J. L.; Cilfone, N. A.; Gulcher, J. R.; Chittenden, T. W. Enhancing Retrosynthetic Reaction Prediction with Deep Learning Using Multiscale Reaction Classification. *J. Chem. Inf. Model.* **2019**, *59*, 673–688.

(20) Watson, I. A.; Wang, J.; Nicolaou, C. A. A Retrosynthetic Analysis Algorithm Implementation. *J. Cheminform.* **2019**, *11*, 1.

(21) Liu, B.; Ramsundar, B.; Kawthekar, P.; Shi, J.; Gomes, J.; Luu Nguyen, Q.; Ho, S.; Sloane, J.; Wender, P.; Pande, V. Retrosynthetic Reaction Prediction Using Neural Sequence-to-sequence Models. *ACS Cent. Sci.* **2017**, *3*, 1103–1113.

(22) Coley, C. W.; Rogers, L.; Green, W. H.; Jensen, K. F. Computer-Assisted Retrosynthesis Based on Molecular Similarity. *ACS Cent. Sci.* **2017**, *3*, 1237–1245.

(23) Coley, C. W.; Jin, W.; Rogers, L.; Jamison, T. F.; Jaakkola, T. S.; Green, W. H.; Barzilay, R.; Jensen, K. F. A Graph-convolutional Neural Network Model for the Prediction of Chemical Reactivity. *Chem. Sci.* **2019**, *10*, 370–377.

(24) Weininger, D. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.

(25) Daylight Theory: SMARTS - a Language for Describing Molecular Patterns. `http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html`.

(26) Landrum, G. RDKit: Open-source Cheminformatics. `http://www.rdkit.org`, Accessed on 2016-11-20.

(27) Cahn, R. S.; Ingold, C.; Prelog, V. Specification of Molecular Chirality. *Angew. Chem. Int. Ed. Engl.* **1966**, *5*, 385–415.

(28) Lowe, D. M. Patent Reaction Extraction: Downloads; Https://bitbucket.org/dan2097/patent-reaction-extraction/downloads. 2014.

(29) Coley, C. W. ASKCOS. http://askcos.mit.edu/.

Supporting Information

RDChiral: An RDKit Wrapper for Handling Stereochemistry in Retrosynthetic Template Extraction and Application

Connor W. Coley, William H. Green*, and Klavs F. Jensen*

Department of Chemical Engineering, MIT, Cambridge, MA

E-mail: whgreen@mit.edu; kfjensen@mit.edu

Code Availability

All code can be found at `https://github.com/connorcoley/rdchiral` in addition to Jupyter notebooks containing all of the examples included in this manuscript.

Additional Results

Template Extraction

The ring closing reaction in Figure S1A, as defined by the reaction SMILES, requires the trans cyclobutane in the precursor. The ketone in Figure S1B is prepared from the chiral alcohol, so the corresponding template specifies a chiral alcohol precursor. Figure S1C defines a reaction between an alkyl iodide and the β position of an α,β-unsatured ketone. Figure S1D defines the preparation of a bromohydrin from a cis epoxide. And, finally, Figure S1E describes the stereoinversion of a chiral mesylate with an azide via an S_N2 reaction.

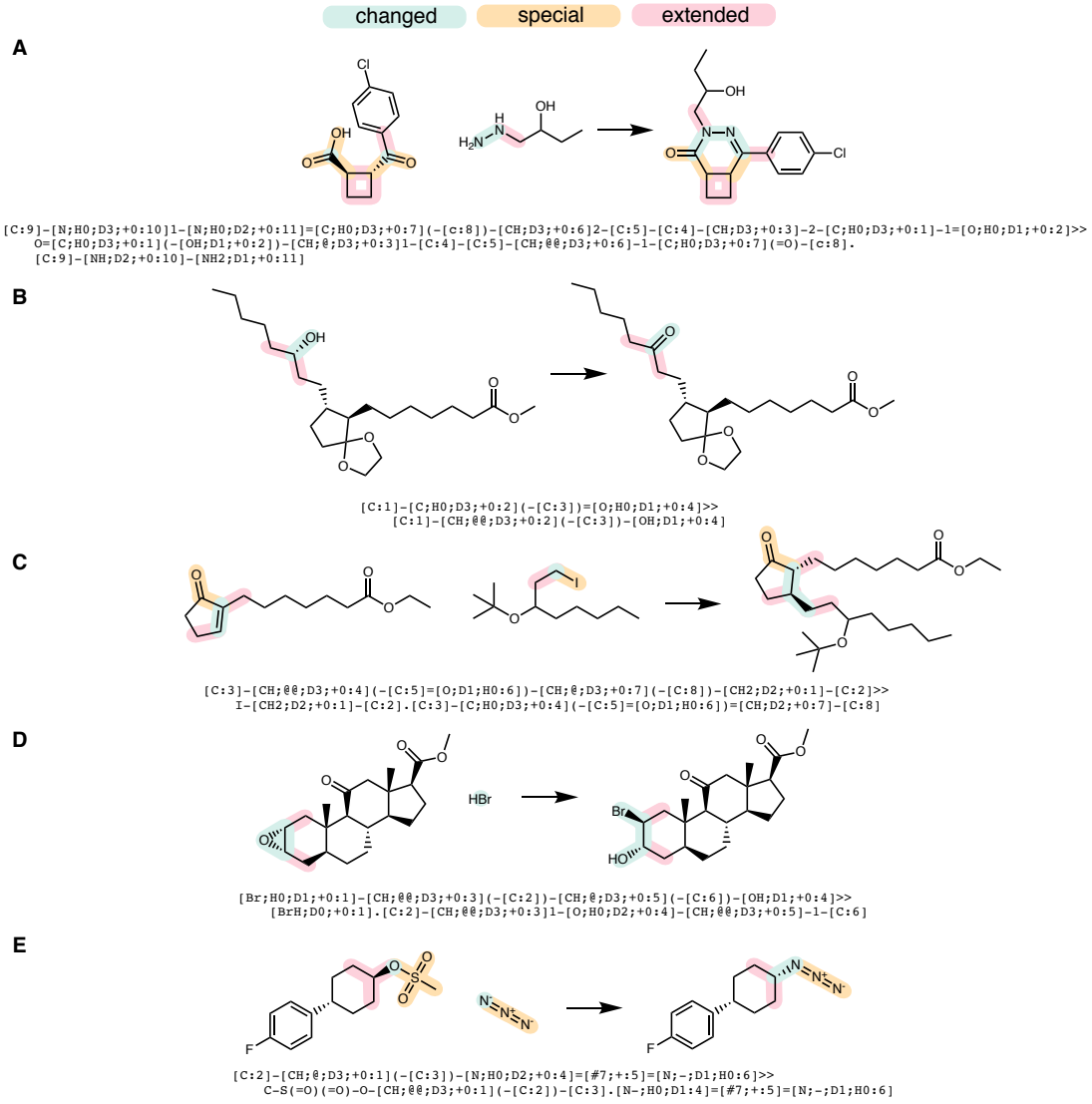

Figure S1: Additional examples of reactions from the USPTO and the retrosynthetic templates we extract from them, where stereochemistry must be considered. Atom mapping is omitted for brevity, but is unambiguous in all cases shown. Spectator molecules are also omitted, as they do not contribute heavy atoms to the product and are not included in the resulting reaction template.

Data Quality Issues

The workflow relies on the availability of atom-mapped reaction SMILES strings. There are cases where the quality of examples can lead to nonsensical reaction templates. Two such examples are shown in Figure S2. The first, shown in Figure S2A, is a result of poor atom mapping. The reaction is a substitution of an alkyl chloride using sodium cyanide to prepare

the alkyl cyanide. However, the solvent dimethylsulfoxide (DMSO) is labeled as contributing to the heterocycle in the product molecule. To the algorithm, all mapped atoms aside from atom 3 appear to undergo a change in connectivity, which causes the resulting template to include the entire product molecule.

The second example, shown in Figure S2B, appears to be an erroneous reaction example entirely. It is likely that the unlabeled molecule was meant to be the reactant and the labeled reactant was meant to be the product of an oxidation. However, given the entry and its atom mapping, the apparent reaction is a shortening of the propyl side chain to ethyl. The corresponding template describes a reaction where any alkyl chain of at least two carbons can be synthesized from the same compound with that chain extended by a single carbon.

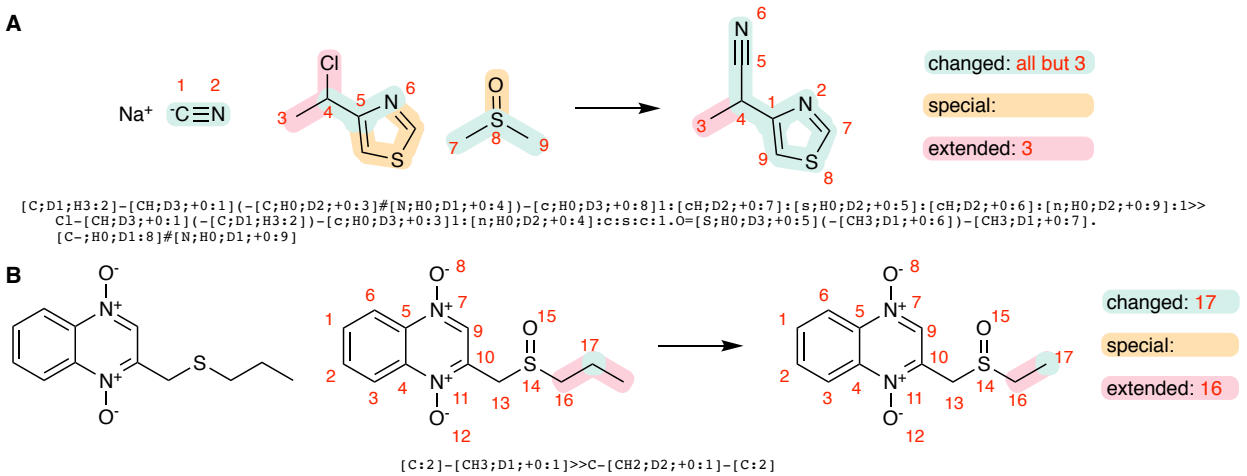

Figure S2: Reactions where the extracted template SMARTS is not chemically meaningful. (A) A case of poor atom mapping, where what should be a simple substitution reaction results in a template that encompasses the entire product structure. (B) A case of poor data quality, where the resulting template suggests that any alkyl chain of length $n \geq 2$ can be prepared from an alkyl chain of length $n + 1$.

Other files

uspto.reactions.json.gz (196.01 MiB)	view on ChemRxiv • download file
uspto.templates.json.gz (50.24 MiB)	view on ChemRxiv • download file