AutoMapper: A Python Tool for Accelerating the Polymer Bonding Workflow in LAMMPS

Matthew A. Bone[a,*], Brendan J. Howlin[b], Ian Hamerton[a], Terence Macquart[a]

[a]*Bristol Composites Institute, Department of Aerospace Engineering, School of Civil, Aerospace, and Mechanical Engineering, University of Bristol, Queen's Building, University Walk, Bristol, BS8 1TR, United Kingdom*
[b]*Department of Chemistry & Faculty of Engineering and Physical Sciences, University of Surrey, Guildford, GU2 7XH, Surrey, United Kingdom*

Abstract

Polymeric materials modelling has the potential to rapidly accelerate the discovery of new materials due to the comparative ease of simulations compared to laboratory testing campaigns. High quality molecular dynamics simulation software, such as LAMMPS, are able to facilitate the transition from empirical to digitised chemistry. However, in order to fully benefit from the speed of simulations, tools need to be developed to automate the preprocessing stages required for modelling. AutoMapper is an open-source Python application that automates the generation of files required to use REACTER, a powerful polymer bonding package implemented within LAMMPS. To automate this process, the authors developed an iterative path search algorithm based on chemical graph theory to accurately map pre- and post-reaction polymerisation structures, and hence eliminating the bulk of the human effort previously required to run a simulation. AutoMapper requires minimal user input, is force field independent, and has shown marvellous performance on a wide range of polymerisation types.

Keywords: Molecular Dynamics, LAMMPS, Polymer Simulation, Open Source Software

1. Introduction

Digital simulation of polymers for materials discovery has a number of advantages over the traditional laboratory test campaign. Molecular dynamics (MD) is a tool that can be used as a cheaper and faster method for screening new materials compared to laboratory campaigns, and produces no physical waste products. MD has been utilised by the pharmaceutical industry to support drug design for the last 40 years [1, 2, 3]. It is expected that the chemical and material industries will follow the pharmaceuticals industry, as seen by the number of large investments made by chemical companies into supercomputing facilities in recent years [4]. The use of MD in industries ranging from agrochemical [5], petrochemical [6], and battery research [7], as well as application in the field of chemical engineering [8], mean MD will remain a useful tool for modern chemistry companies. A range of high quality, open-source MD codes, such as NAMD [9], GROMACS [10], and LAMMPS [11], are available to support this industry, however further developments into accompanying software, such as the ones presented herein, are required to aid in adoption, and fully leverage, the technology.

There are many examples in literature of polymer modelling using a "stop-start" procedure to create a cured structure from individual monomers [12, 13, 14, 15]. The typical workflow involves creating a series of bonds, based on suitable neighbouring atoms within a cutoff distance, followed by structure relaxation using geometry

*Corresponding author.
Email address: matthew.bone@bristol.ac.uk (Matthew A. Bone)

optimisation, extended dynamics simulations, or both. This is then repeated for multiple rounds of bonding, stopping dynamics to create bonds and restarting dynamics to equilibrate the structure, until a cured polymer unit cell has been created. Whilst this methodology has been shown to be effective on a range of polymer systems, it suffers from high computational cost as the whole simulation cell must be relaxed once bonds are formed. The alternative REACTER algorithm implemented by Gissinger *et al.* streamlines the polymer bonding process by providing a robust algorithm capable of modelling both simple and complex bonding reactions alike [16, 17]. This is a great improvement on many user based polymerisation scripts which are often bespoke to a particular polymer. REACTER uses pre- and post-reaction molecule templates to determine where bonds can form and how the surrounding topology changes in the post-bond molecule. Bonds are formed by finding valid bonding pairs within a cutoff distance, and adapting the structure between two timesteps of the dynamics simulation allowing for seamless creation of a polymer structure without ever halting the dynamics. Reacted atoms are isolated and the initial geometry relaxed using a microcanonical NVE ensemble (wherein the system is isolated from changes in number of particles (N), volume (V), and energy (E)) with motion limits for user specified amount of time, whilst the rest of the simulation box continues to model atomic motion with the originally specified ensemble.

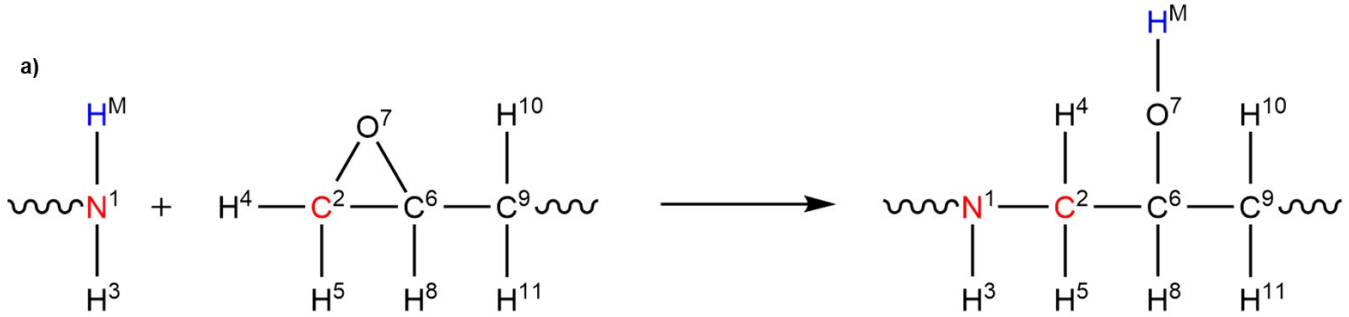
REACTER suffers a key drawback which is the challenge of creating pre- and post-reaction molecule templates, along with a map file that describes how the atom IDs in each file relate to one another. The process of generating these templates is time consuming and error prone due to the poor human readability of the LAMMPS data files from which the templates are derived. Within this context, the present paper showcases AutoMapper, an open-source Python toolbox designed to automate the preprocessing workflow required to use REACTER using LAMMPS input data files and minimal user input. In this work, a LAMMPS input data file is the text file that defines all the atoms in a simulation, the atom positions within a unit cell, and the connectivity of the atoms to one another. Mapping atoms across chemical reaction is a common problem within chemistry, and many recent papers have explored the use of machine learning to automate the mapping and classification of complex and unseen chemical reactions [18, 19]. By contrast, AutoMapper tackles a simplified problem as it only seeks to map polymerisation reactions, yet can achieve excellent mapping performance with a streamlined algorithmic alternative to machine learning models.

The aim of AutoMapper is to eliminate human error from generation of REACTER input files and enable higher throughput MD polymer modelling, by creating input files significantly faster than manual attempts, to support data science techniques such as machine learning. REACTER is implemented in LAMMPS as *fix bond/react* and will be referred to from here as such. AutoMapper has been designed to be force field independent and shown to work with a multitude of polymer systems. These range from simple molecules to more complex multi-stage reactions. The following sections detail the core mechanics of AutoMapper, which is based around a custom path search developed for this work, and give examples of different molecular environments that can be mapped with AutoMapper. AutoMapper can be downloaded from GitHub at https://github.com/m-bone/AutoMapper which includes a manual, tutorials and many test example polymers, explaining how to use the software.

2. Software Philosophy and Design

At its core, AutoMapper is a chemistry driven path search which utilises chemical graph theory, whereby atoms are declared as nodes and bonds as edges [20]. Modelling molecules as a graph allows the path search to make use of the fact that the majority of the connectivity structure of a molecule remains unchanged by a reaction. This methodology is inspired by the superimpose algorithm used within *fix bond/react* [16]. Two graphs of the molecules pre- and post-reaction are

a)



b)

Map Stage (Start Atom)	Post Mapped Structure	List of Mapped Pair IDs	Path Search Queue IDs	Summary
Initialisation (User Specified)	$N^1—C^2$	1, 2	1, 2	User specified bonding atoms begin queue.
1 (Atom 2)	H^4, $N^1—C^2—C^6$, H^5	1, 2, 4, 5, 6	1, 6	Carbon 6 found by single element occurrence. Hydrogens mapped randomly.
2 (Atom 6)	H^4, O^7, $N^1—C^2—C^6—C^9$, H^5, H^8	1, 2, 4, 5, 6, 7, 8, 9	1, 7, 9	All atoms found by single element occurrence as they are all unique elements.

Figure 1: How the path search evolves is shown with an example part of an epoxidation reaction (a) with the bonding atoms marked in red, and all atoms given an ID value. The hydrogen in blue with ID "M" is found through the missing atom function. The table (b) shows how an initial bonding atom pair supplied by the user grows into a fully mapped structure. For simplicity, in this instance pre- and post-reaction ID values match, but this is not always true.

explored to determine which nodes represent the same atoms between the molecules, as illustrated in Figure 1.

The challenge when identifying molecule templates by hand is that the atom IDs do not correspond i.e. pre-reaction ID 1 is not necessarily post-reaction ID 1. There are many factors that determine this ID assignment, so it is not possible to build a heuristic solution to ID matching. Instead, AutoMapper uses the explicitly identified atoms involved in creating new bonds, the "bonding atoms", to initialise the graph building process. From this known starting point, it is possible map pre- and post-reaction atom pairs together, regardless of ID. In many cases this can be achieved by looking at the number of times an element occurs in the neighbours of a known atom. Other techniques have been developed to map pairs when this simple check is not possible, such as declaring an atom "missing" so that it may be found after the initial mapping process is complete. Iterating through the graph using the previously mapped pairs as a starting point for comparison allows the map to grow gradually. Once complete, a list of pre- and post-reaction atom ID pairs are output and formatted for the *fix bond/react* map file. This system eliminates the need for any force field information, which is vital to the flexibility of AutoMapper. This methodology is extensible: by using AutoMapper multiple times with different pre- and post-reaction structures it is possible to create all the molecule files required for complex multi-stage reactions.

In order to improve upon the manual generation of files for *fix bond/react*, the design for AutoMapper required as little user input as possible. As a result, AutoMapper inputs are limited to:

- Pre-reaction LAMMPS input data file;
- Post-reaction LAMMPS input data file;

- A list of elements by LAMMPS atom type;
- Atom IDs of the bonding atoms;
- Atom IDs of atoms to be deleted after reaction (optional).

The LAMMPS input data files of the pre- and post-reaction molecule must only include one set of the molecules required for a reaction. These files can be generated simply using tools for molecule editing (e.g. Avogadro [21]) and LAMMPS input data file generators such as Moltemplate [22]. The user may face some challenges in identifying the atom IDs of the bonding atoms due to the complex nature of large LAMMPS input data files. However, as these atoms define the initial start point for the path search, they are vital to the operation of the algorithm. Including atoms to be deleted is optional as this is typically only done with byproducts that the user doesn't wish to represent in the final bonded structure of the polymer e.g. water molecules formed by condensation polymerisation. Identified atom IDs for deletion simplify the mapping process as these atom IDs define another starting point for the path search. Additionally, there are tools in place to identify byproducts that have not been flagged for deletion, so that these atoms are mapped correctly should the user wish to keep the byproducts. This list of required information should be readily available to the user as much of it is already needed for a standard LAMMPS simulation using *fix bond/react*.

A high level overview of the AutoMapper process is seen in Figure 2. The LAMMPS input data files are converted to the molecule file format and output as these files are required for *fix bond/react*. The molecule files and the bonding atom IDs are then passed to the mapping function seen in Figure 3. Once a full map is created, AutoMapper determines if an acceptable partial structure of the molecule can be found. A partial structure is the fewest number of atoms required to build a reactive site: the environment around the atoms that have just bonded. A partial structure is used to maintain the validity of the molecule templates as a polymer chain increases in length: otherwise a template would be required for each possible chain length. Equally, the pattern matching algorithm used by *fix bond/react* works faster on smaller patterns, so there is a computational advantage to using the smallest possible partial structure. If a partial structure can be found, the map and molecule files are reduced to reflect the new structure, and output to the user. Otherwise, the map is output without further modification.

Returning to the mapping process described in Figure 3, the user supplied bonding atoms are used to initialise the queue and begin the path search process to generate a map, as discussed above and seen in Figure 1. To determine which pre- and post-reaction atoms are paired, the neighbours of a known atom are compared. In the first instance the known atom is one of the bonding atoms but, with each successful map of a non-hydrogen atom, the newly mapped atom pair enters the queue to become the starting atom for the next stage of comparison. This is depicted in the stages of Figure 1(b) where the carbon atom with ID 6 is found from carbon ID 2, which in turn is used to continue the search and determine atom IDs 7, 8 and 9. Should a comparison fail to determine a suitable match for a given atom, the atom is moved to a "missing atom" list where it is handled after the path search queue has been cleared. A more detailed explanation of this can be found in Section 3.2. With the missing atoms solved the resultant full map is passed through the partial structure check to be output, as seen in Figure 2.

To facilitate the creation of a partial structure, *fix bond/react* utilises "edge" atoms to distinguish where the relevant partial structure ends. The region around the newly formed bond, the reactive site, must contain all the atoms that could be affected by the new bond formation. As most force fields include the four body dihedral and improper dihedral terms, a minimum reactive site must include all atoms three bonds away from the bonding atoms, as seen in Figure 4. This must be extended if other atoms in the reactive site change force field type during the reaction, as these changes may require different force field co-

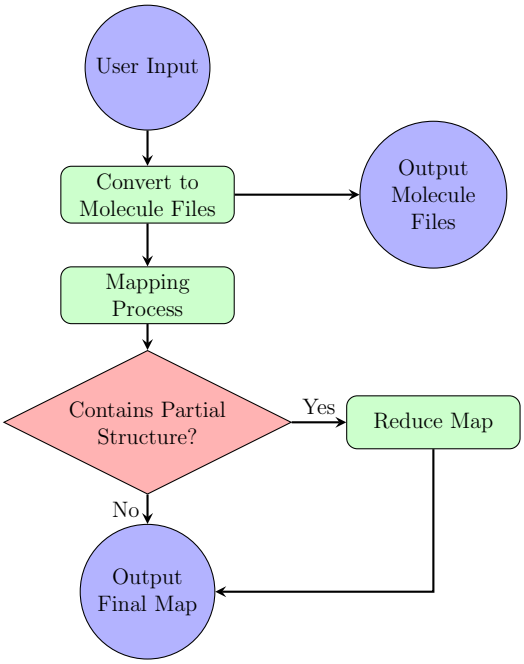


Figure 2: Flowchart showing a high level overview of the main mapping process within AutoMapper. The "Mapping Process" node is expanded upon in Figure 3.

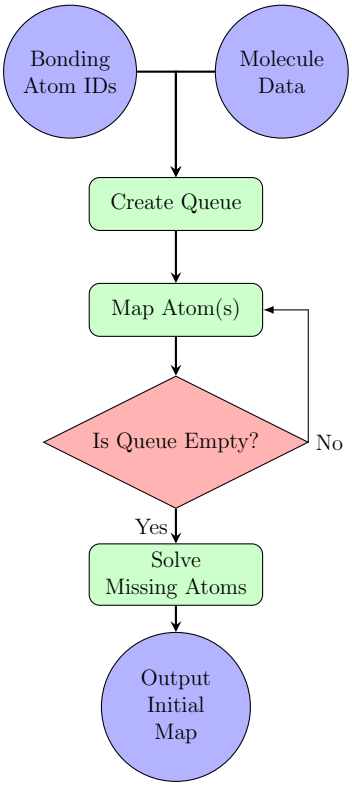


Figure 3: Flowchart of the **mapping process**. Once initiated, the queue is self-sustaining using the mapped atoms of the prior output as the input for the next iteration. The output of this flowchart feeds into the "Mapping Process" node of Figure 2. The "Map Atom(s)" node is expanded upon in Figure 5.

efficients for bond/angle/dihedral/improper connections in other parts of the polymer structure. AutoMapper establishes the smallest possible partial structure by reducing the map down considering these reactive site rules, and assigning edge atoms as appropriate.

3. Key Functions in AutoMapper

3.1. Molecule File Converter

The template files used in *fix bond/react* must be in the LAMMPS "molecule" format which is subtly different to typical LAMMPS input data files, and take a long time to assemble manually. The key difference between the formats is LAMMPS input data files can be found in a range of different "styles", whilst the LAMMPS molecule files are a fixed style that explicitly defines key atom properties. To alleviate the conversion effort, a tool was written to read LAMMPS input data files in the "full" style that represent the molecules required for a reaction. The files were then automatically converted to the LAMMPS molecule file format and output for the user to use in *fix bond/react* simulations. This

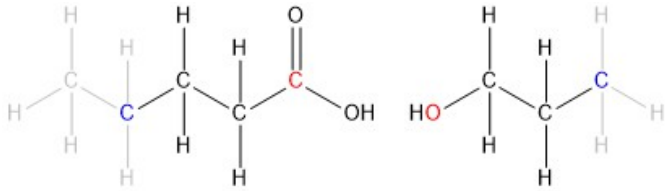


Figure 4: Structure highlighting the bonding atoms (red) and edge atoms (blue) for an esterification reaction. The greyed out atoms are not required in the reactive site as they are not involved in the interactions of the bonding atoms, which are changing force field type due to the reaction.

is then built into the mapping process (the first stage in Figure 2) as a convenience for the user and eliminates the human error associated with creating these files by hand. This tool can also operate standalone from the mapping function in AutoMapper to allow users to generate molecule format files for other LAMMPS functions that re-

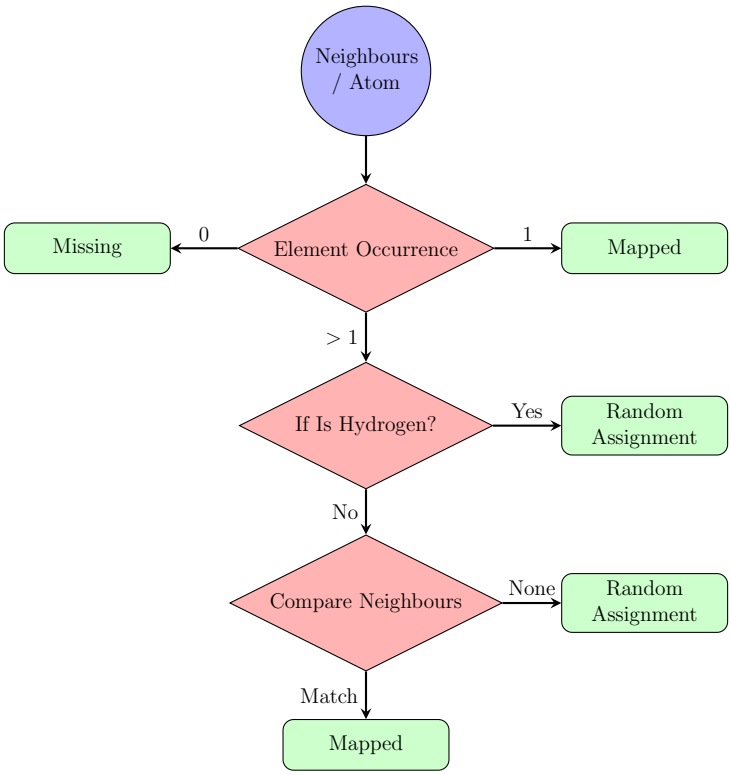


Figure 5: Flowchart of the decision making process used to map atoms. For the initial atom mapping stage, the neighbours of an atom are evaluated. For missing atoms, the missing atom itself is evaluated against all other possible missing atoms. This process forms the basis of the "Map Atom(s)" stage in Figure 3.

quire them.

3.2. Map Decision Making

To compare pre- and post-reaction atoms to one another a robust decision making protocol has to be written. The flowchart seen in Figure 5 details this process, which is used in both the initial mapping stages and to resolve missing atoms. For the initial mapping stage the neighbours of known pre- and post-reaction atom pair are compared: in the first instance of the queue these atoms are the user specified bonding atoms. For missing atoms a pre-reaction missing atom is compared to all available post-reaction missing atoms. The element occurrence check in Figure 5 counts the number of atoms for a given element present in the available neighbours. The assumption is made that if only one carbon atom is present in the pre- and post-reaction neighbour list, it is the same carbon atom in each instance and a map is made. This simple check resolves the majority of non-hydrogen mappings. If an element only appears in the pre-reaction neighbour list, and not in the post-reaction neighbours, the atom is declared missing and is mapped later. If an element occurs more than once it cannot be mapped simply and passes down the decision tree.

At the next stage, hydrogen atoms that appear more than once are assigned randomly from the pool of available hydrogen atoms. Hydrogen atoms are assumed to only form one bond and therefore cannot be misassigned as nothing relies on bonding through a specific hydrogen. This is a limitation as it does not allow for borane style bonding, however, it was decided this is a niche use case that could not be catered for without requiring more information from the user. When mapping any other multiple occurrence elements the neighbours of the atom in question are considered. The neighbours of the atom are combined to create an alphabetised string referred to as a "fingerprint". For example, an atom bound to two hydrogen atoms, a carbon atom, and an oxygen atom would have a fingerprint of "CHHO". These fingerprint strings are then compared to

fingerprints of all the possible atom pairs and a unique string match will lead to a successful map. This is tried for the first neighbours of an atom, but if no unique match can be found the process is retried for the second and third neighbours. If no successful string match can be found, the atom environment is clearly highly symmetric and assignment can be made at random without affecting the quality of the map. The assumption is if the two atoms are symmetric to the third neighbours, the force field coefficients defining the bond/angle/dihedral/improper interactions are also the same. A warning is raised for the user when this occurs so that they may check the map is suitable, but testing failed to find an instance where this was not suitable.

3.3. Ring-Opening Polymerisation

As discussed in Section 2, a full map is reduce to the smallest valid representation of the reactive site, as these partial structures have many practical benefits. Determining these partial structures is typically trivial as identifying all atoms within three bonds of the bonding atoms, and checking for force field type changes, is achieved by iterating through the neighbours of the bonding atoms. However, for a large ring opening polymerisation, such as the polycaprolactam displayed in Figure 6, there are additional challenges. As the ring opens, the nitrogen atom adjacent to the bonding atoms changes from being one bond away to six bonds away. This places the nitrogen atom outside of the necessary three bonds required to keep dihedral/improper interactions rendered within the partial structure. However, as the nitrogen is found in the pre-reaction structure, it must be included in the post-reaction structure as *fix bond/react* requires the pre- and post-reaction molecule files be the same length. To maintain all the ring atoms in the pre- and post-reaction structure, AutoMapper uses a breadth first path search to determine if the pre-reaction structure is a ring, and if this ring has been broken in the post-reaction structure. All atoms involved in the ring are then identified and maintained in the post-reaction partial structure, with edge atoms created off of the ring atoms as required. This addition facilitates any large ring opening polymerisation reactions the user may wish to model.

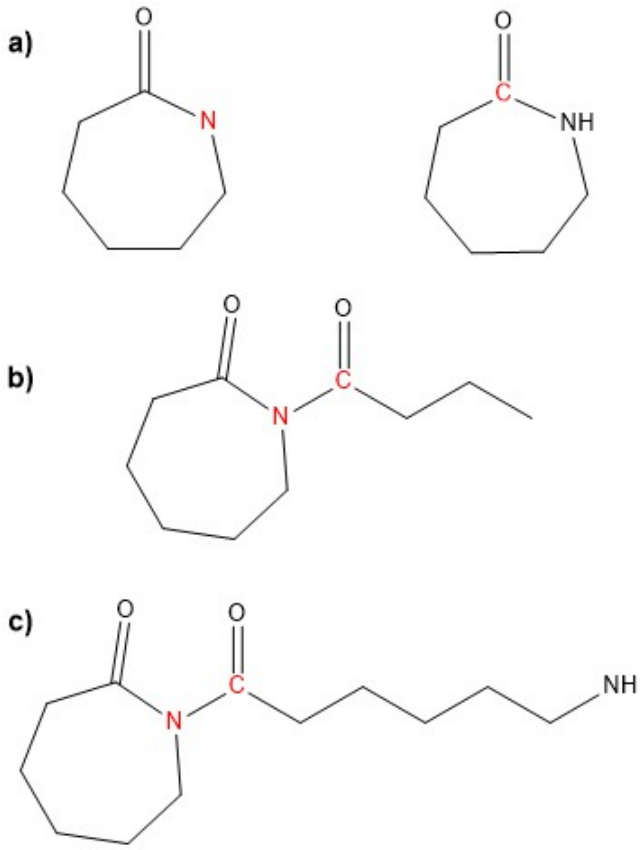


Figure 6: Partial structures used in the initiation of a ring opening polymerisation of caprolactam. Bonding atoms are highlighted in red. a) The pre-reaction partial structure. b) The post-reaction partial structure without considering the ring-opening. This structure is invalid as it has fewer atoms than a). c) The post-reaction partial structure found by the ring-opening function.

4. Mapped Examples

To demonstrate the different algorithms within AutoMapper, the polymerisation of a polyacrylate, and phenolic resin are used to highlight the systems at work. Hydrogen atoms are ignored in these examples as they are either assigned with single element occurrence, or assigned randomly from a pool of hydrogen atoms connected to the same atom. The polyacrylate example in Figure 7 is resolved using a mixture of single element occurrence and comparison of first neighbour strings. Beginning at the highlighted user supplied bonding atoms, the first two carbon

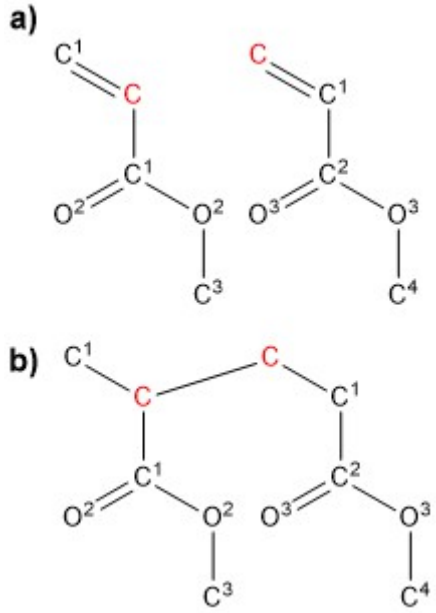


Figure 7: Schematic of the mapping order for the polymerisation of a acrylate monomers with a) pre-reaction and b) post-reaction molecules. Bonding atoms are highlighted in red are the starting point for the map. Hydrogen atoms have been omitted for clarity.

atoms in the left hand side molecule are differentiated by comparing the first neighbour fingerprint strings. Despite the double bond being reduced to a single bond, the carbon involved in the ester remains unchanged and can be identified by the fingerprint "COO". The other carbon is found afterwards as there is now only one unassigned carbon atom in the pool. The acrylate oxygen atoms are found much the same way as the carbonyl oxygen has a fingerprint of "C" whilst the ester oxygen is "CC". The ester carbon is a single carbon occurrence in the pool, and the process is repeated for the right hand side molecule.

The phenolic resin example in Figure 8 is more complicated than the polyacrylate as the water byproduct needs to be mapped using the missing atoms search. Starting at the marked bonding atoms, the rings are found by using a mixture single element occurrence and first neighbour checks. The hydroxyl group acts as a useful indicator to differentiate between the aromatic carbons. If no indicator was present, the carbons found as the map enters the ring (the C1 atoms on the right, the C2 atoms on the left) would have to be assigned randomly. The missing atoms are resolved after the map is complete by looking at what atoms remain unpaired. The OH and H atoms in blue will be missed in the mapping process as they are not in the main structure of the post-reaction molecule. As only one hydrogen and oxygen are missing the decision making is a simple single element occurrence check. The hydrogen bound to oxygen in the OH won't have been declared missing, and will be found once the oxygen has been mapped.

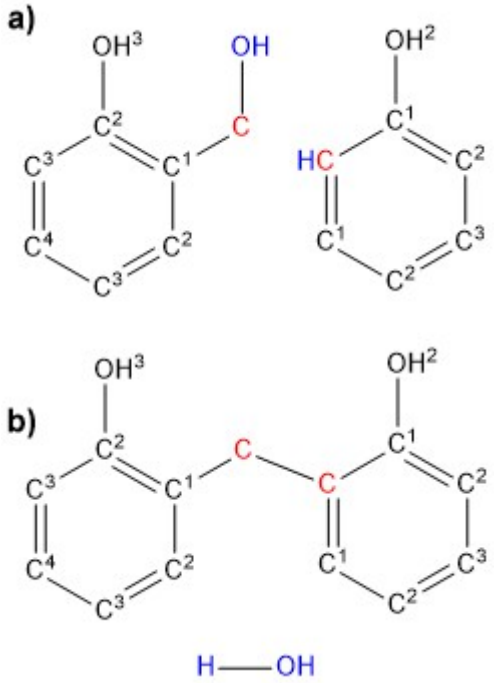


Figure 8: Schematic of the mapping order for the bonding of a phenolic resin with a) pre-reaction and b) post-reaction. Bonding atoms are highlighted in red are the starting point for the map. Missing atoms are seen in blue and are found after the rest of the map by pairing together all the atoms that could not be initially mapped. Most hydrogen atoms have been omitted for clarity.

5. Conclusions

The potential for rapid materials discovery through digital materials design drives the need for tools that automate the modelling process. AutoMapper fulfills an aspect of this need by eliminating the time consuming and error prone process of mapping pre- and post-reaction templates, a requirement for using the powerful poly-

mer bonding tool *fix bond/react* in LAMMPS. This work highlights how AutoMapper is able to be force field independent, and capable of mapping a broad spectrum of polymerisation reactions. The methodology used by AutoMapper has the potential to be applied to other problems, such as identifying the force field coefficients of an atom. The tools described herein can be coupled with other preprocessing tools to form the basis of an automated workflow for high throughput modelling. A suite of mature automation tools will enable high throughput MD modelling that can leverage the advances in data science and machine learning, and help accelerate the materials design process.

Acknowledgments

This research received no external funding. M. A. Bone is supported by the Engineering and Physical Science Research Council as part of the EPSRC Centre for Doctoral Training in Composites Science, Engineering, and Manufacturing. Grant number: EP/S021728/1. M. A. Bone would like to thank Jacob Gissinger, Andrew Jewett and Benjamin Jenson for their useful discussion around the development of AutoMapper.

Data availability

The raw data required to reproduce these findings are available to download from https://github.com/m-bone/AutoMapper.
The processed data required to reproduce these findings are available to download from https://github.com/m-bone/AutoMapper.

References

[1] A. Ganesan, M. L. Coote, K. Barakat, Molecular dynamics-driven drug discovery: leaping forward with confidence, Drug Discovery Today 22 (2) (2017) 249–269. doi:10.1016/j.drudis.2016.11.001.

[2] H. Zhao, A. Caflisch, Molecular dynamics in drug design, European Journal of Medicinal Chemistry 91 (2015) 4–14. doi:10.1016/j.ejmech.2014.08.004.

[3] M. De Vivo, M. Masetti, G. Bottegoni, A. Cavalli, Role of Molecular Dynamics and Related Methods in Drug Discovery (5 2016). doi:10.1021/acs.jmedchem.5b01684.

[4] R. Mullin, Digitalization comes to the materials industry, Chemical & Engineering News 95 (39) (2017) 26–30.

[5] H. A. Oyewusi, F. Huyop, R. A. Wahab, Molecular docking and molecular dynamics simulation of Bacillus thuringiensis dehalogenase against haloacids, haloacetates and chlorpyrifos, Journal of Biomolecular Structure and Dynamics (2020) 1–16doi:10.1080/07391102.2020.1835727.
URL https://doi.org/10.1080/07391102.2020.1835727

[6] S. Yaseen, G. A. Mansoori, Asphaltene aggregation due to waterflooding (A molecular dynamics study), Journal of Petroleum Science and Engineering 170 (2018) 177–183. doi:10.1016/j.petrol.2018.06.043.

[7] M. T. Ong, O. Verners, E. W. Draeger, A. C. Van Duin, V. Lordi, J. E. Pask, Lithium ion solvation and diffusion in bulk organic electrolytes from first-principles and classical reactive molecular dynamics, Journal of Physical Chemistry B 119 (4) (2015) 1535–1545. doi:10.1021/jp508184f.

[8] E. J. Maginn, J. R. Elliott, Historical perspective and current outlook for molecular dynamics as a chemical engineering tool, Industrial and Engineering Chemistry Research 49 (7) (2010) 3059–3078. doi:10.1021/ie901898k.

[9] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kalé, K. Schulten, Scalable Molecular Dynamics with NAMD, Journal of Computational Chemistry 26 (16) (2005) 1781–1802. doi:10.1002/jcc.20289.

[10] H. J. C. Berendsen, D. van der Spoel, R. van Drunen, GROMACS - A Message-Passing Parallel Molecular-Dynamics Implementation, Computer Physics Communications 91 (1-3) (1995) 43–56. doi:10.1016/0010-4655(95)00042-e.

[11] S. Plimpton, Fast Parallel Algorithms for Short-Range Molecular Dynamics, Journal of Computational Physics 117 (1) (1995) 1–19. doi:10.1006/jcph.1995.1039.

[12] M. A. Bone, T. Macquart, I. Hamerton, B. J. Howlin, A Novel Approach to Atomistic Molecular Dynamics Simulation of Phenolic Resins Using Symthons, Polymers 12 (4) (2020) 926. doi:10.3390/polym12040926.

[13] Y. Shudo, A. Izumi, K. Hagita, T. Nakao, M. Shibayama, Large-scale molecular dynamics simulation of crosslinked phenolic resins using pseudo-reaction model, Polymer 103 (2016) 261–276. doi:10.1016/j.polymer.2016.09.069.

[14] B. Demir, T. R. Walsh, A robust and reproducible procedure for cross-linking thermoset polymers using molecular simulation, Soft Matter 12 (8) (2016) 2453–2464. doi:10.1039/c5sm02788h.

[15] M. S. Radue, V. Varshney, J. W. Baur, A. K. Roy,

G. M. Odegard, Molecular Modeling of Cross-Linked Polymers with Complex Cure Pathways: A Case Study of Bismaleimide Resins, Macromolecules 51 (5) (2018) 1830–1840. doi:10.1021/acs.macromol.7b01979.

[16] J. R. Gissinger, B. D. Jensen, K. E. Wise, Modeling chemical reactions in classical molecular dynamics simulations, Polymer 128 (2017) 211–217. doi:10.1016/j.polymer.2017.09.038.

[17] J. R. Gissinger, B. D. Jensen, K. E. Wise, Reacter: A heuristic method for reactive molecular dynamics, Macromolecules 53 (22) (2020) 9953–9961. doi:10.1021/acs.macromol.0c02012.

[18] W. Jaworski, S. Szymkuć, B. Mikulak-Klucznik, K. Piecuch, T. Klucznik, M. Kaźmierowski, J. Rydzewski, A. Gambin, B. A. Grzybowski, Automatic mapping of atoms across both simple and complex chemical reactions, Nature Communications 10 (1) (3 2019). doi:10.1038/s41467-019-09440-2.

[19] P. Schwaller, D. Probst, A. C. Vaucher, V. H. Nair, D. Kreutter, T. Laino, J.-L. Reymond, Mapping the space of chemical reactions using attention-based neural networks, Nature Machine Intelligence 3 (2) (2021) 144–152. doi:10.1038/s42256-020-00284-w.
URL https://doi.org/10.1038/s42256-020-00284-w

[20] K. J. Burch, Chapter 8 - Chemical applications of graph theory, in: S. M. Blinder, J. E. House (Eds.), Mathematical Physics in Theoretical Chemistry, Developments in Physical & Theoretical Chemistry, Elsevier, 2019, pp. 261–294. doi:10.1016/B978-0-12-813651-5.00008-5.

[21] M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, G. R. Hutchison, Avogadro: an advanced semantic chemical editor, visualization, and analysis platform, Journal of Cheminformatics 4 (2012) 17. doi:10.1186/1758-2946-4-17.

[22] A. I. Jewett, D. Stelter, J. Lambert, S. M. Saladi, O. M. Roscioni, M. Ricci, L. Autin, M. Maritan, S. M. Bashusqeh, T. Keyes, R. T. Dame, J. E. Shea, G. J. Jensen, D. S. Goodsell, Moltemplate: A Tool for Coarse-Grained Modeling of Complex Biological Matter and Soft Condensed Matter Physics, Journal of Molecular Biology (2021). doi:10.1016/j.jmb.2021.166841.