UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

An efficient simulation of crosslinked RAFT copolymerization

P.A.H. van Dijk

M.Sc. Thesis
May 2018

Supervisors:
prof. dr. ir. A. Rensink
prof. dr. ir. H.J. Broersma
dr. ir. J. Paulusse
M.R. Elzes, MSc.

Formal Methods & Tools Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Abstract

Crosslinked RAFT copolymerization, a process with which nanometer-sized polymer networks are created, is relatively new and as such many experiments are needed to further develop this technique. These experiments are both resource and time consuming, making this an expensive undertaking. The aim of this project was the simulation of this polymerization process in order to make predictions about the effects of the monomer concentration, the crosslinker/monomer ratio and the RAFT agent/monomers ratio on the size, weight distribution and crosslink density of the resulting polymers. Predictions of crosslink density are especially interesting, since this parameter is difficult to quantify experimentally. Such a simulation can thus not only save time and money, but also give more insight into the polymerization process.

We chose to create a Kinetic Monte Carlo (KMC) simulation. Normally, a KMC simulation keeps track of the reaction rates of all possible reactions in the simulation. However, the number of possible reactions scales almost quadratically with the number of different molecules when simulating crosslinked polymerization, making the regular KMC algorithm unfit for large simulations. To overcome this problem, the tracking of the reaction rates was replaced by the tracking of the reactivity of the molecules in the simulation. The efficiency of the simulation was further improved by tracking these reactivities using Binary Indexed Trees (BITs), which provide both efficient time and memory scaling. The KMC algorithm was further extended with the tracking of polymer structures, which was used for providing visual feedback.

The results of the simulation predicted increases in weight differences and polymer size for increases in monomer concentration, decreases in the ratio of RAFT agent to monomer and increases of the ratio of crosslinker to monomer. These predictions are in good correspondence to lab data. Additionally, higher crosslink density was predicted for higher ratios of crosslinker to monomer.

The created simulation scales almost linearly in time with the number of initial molecules in the simulation. Furthermore, the memory usage scales linearly with the number of different molecule species in the simulation. The techniques used in the simulation algorithm are not limited to crosslinked RAFT copolymerization and can also be applied to speed up other simulations of (networked) polymerization. Overall, the created simulation is useful in efficiently predicting a wide range of polymer properties and can be used to support or even replace lab experiments.

Acknowledgements

I want to thank the following people for their help and guidance during the project.

Dr. ir. Jos Paulusse, thank you for hosting me in your research group. Your enthusiasm and curiosity kept me motivated throughout the project and made me get far more out of the research than I would have achieved without. Furthermore, I would like to thank you for making a room available for me in the faculty in which I was able to evade the summer heat and comfortably work on the research.

Prof. dr. ir. Arend Rensink, thank you for your help in writing this thesis. Your feedback was both plentiful and helpful and my writing has improved greatly with your help. Also, I would like to thank you for your patience and help in making critical design decisions.

Prof. dr. ir. Hajo Broersma, thank you for your help during my research. You always got me in contact with the people with the necessary expertise and helped to successfully launch the collaboration between the two faculties.

Rachèl, thank you for your daily supervision during the project. Without you I would not have ended up doing the research I did, as it was your idea and initiative that landed me in the Jos Paulusse Research Group. Furthermore, I would like to thank you for always being available for questions and for the insightful discussions that followed from those. (Sorry for those late-night ones!) I hope we continue our discussions for many years to come.

I want to thank all the members of the Jos Paulusse Research Group. I enjoyed my time with you and had a lot of fun during our activities.

I also want to thank the members of IAPC, who allowed me to more than once fill their entire whiteboard with my calculations, code and molecule structures. Discussions with you were always interesting and led to many new ideas. I want to thank Lars Corbijn van Willenswaard in particular, who introduced me to Fenwick Trees and helped to greatly improve the mathematical notations in my report. Your critical eye is always appreciated, both in coding and writing.

Finally, I want to thank my friends and family, who where always supportive during my research, even when I was so busy that I had little time for them. Your support is greatly appreciated and got me through many stressful moments.

Contents

Nomenclature

Concentration: Unless stated otherwise, concentration is the number of molecules divided by volume of mixture, given in moles per liter ($\mathrm{mol\,l^{-1}}$).
Conversion: The percentage of vinyl groups that has reacted.
Crosslinker: Difunctional monomer, i.e. a monomer with two reactive groups, symmetric in nature.
Crosslinking: Reaction between a radical and a pendent vinyl group, either within that same radical (intramolecular) or in another molecule (intermolecular).
Crosslink site: Point at which a crosslinker is connected to a polymer chain.
Cyclization: Intramolecular crosslinking.
Half-initiator: One half of an initiator.
Initiation: Reaction between an initiator radical and a vinyl.
Initiator: Molecule that is split in half to from two initiator radicals.
Initiator radical: A radical half-initiator.
Intermolecular reaction: Reaction between two molecules.
Intramolecular reaction: Reaction between atoms in the same molecule.
Mole: Unit of measurement for amount of molecules, with the unit symbol mol. A mole is approximately $6.022140857 \times 10^{23}$ molecules [1].
Molecule: Initiator radicals, monomers, crosslinkers and polymers.
Monomer: Monofunctional monomers, i.e. a monomer with a single reactive group. This does not include multifunctional monomers like crosslinkers, unless specifically stated otherwise.
Pendent vinyl group: Active vinyl group in a polymer.
Product: The molecules that are formed during a reaction.
Propagation: Reaction between a polymer and a mono- or multifunctional monomer.
Radical: Molecule containing an unpaired electron, also knows as a free radical electron.
Reactants: The molecules that are consumed during a reaction.
Reactive center: Site of the free radical electron within a molecule. The reactive center will be referred to as such, even when a RAFT agent is attached to this site.
Termination: Reaction between two radicals, forming a non-radical product.
Vinyl: Molecule containing a vinyl group.
Vinyl group: Atom group containing two double bonded carbon atoms.

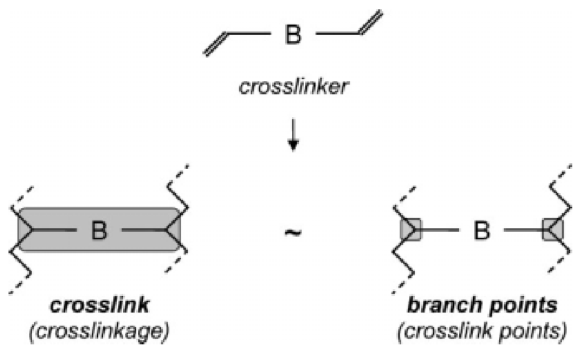


Figure 1: Nomenclature used in this work [2].

Abbreviations

BIT: Binary indexed tree.
CC: Crosslinked copolymerization.
CRP: Controlled (free) radical polymerization.
DLS: Dynamic Light Scattering.
EGDMA: ethylene glycol dimethacrylate.
FRP: Free radical polymerization.
GMA: Glycidyl methacrylate.
GPC: Gel Permeation Chromatography.
KMC: Kinetic Monte Carlo.
MWD: Molecular weight distribution.
ODE: Ordinary differential equations.
PABTC: 2-propanoic acid butyl trithiocarbonate.
PDI: (Mass) Polydispersity Index.
RAFT: Radical Addition-Fragmentation chain-Transfer.
RKMC: Rejection Kinetic Monte Carlo.
SMA: Solketal methacrylate.
SPDI: Size Polydispersity Index.

Parameters and notations

Parameter	Description
I	Initiator radical molecule species.
M	Monomer molecule species.
C	Crosslinker molecule species.
P_i	Polymer molecule species with identifier $i \in \{4, 5, ..., K\}$. Polymer $P_i = \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i\rangle$ consists of I_i half-initiator molecules, M_i monomers and C_i crosslinkers. Furthermore it has V_i pendent vinyl groups, EN_i chain end non-crosslinker radicals, EC_i chain end crosslinker radicals and MC_i mid-chain crosslinker radicals. Note that $V_i = v_i$ and $Z_i = r_{Z,i}$ for each $Z \in \{EN, EC, MC\}$. A different notation is used for readability, making it clearer when we refer to elements of the tuple representing a molecule species.
S_i	Molecule species with identifier $i \in \{1, 2, ..., K\}$. Either an initiator radical, monomer, crosslinker or polymer, i.e. $S_i \in \{I, M, C, P_1, ..., P_{K-3}\}$.
c_i	Number of molecules of species S_i.
Z	Position of a radical group. Either chain end non-crosslinker molecule (EN), chain end crosslinker molecule (EC) or mid-chain crosslinker molecule (MC), i.e. $Z \in \{EN, EC, MC\}$.
r_i	Number or radical groups in molecule species S_i.
$r_{Z,i}$	Number of radical groups of molecule of species S_i in position Z.
r_T	Total number or radical groups in simulation. Defined as $\sum_{i=1}^{K} c_i r_i$
v_i	Number or vinyl groups in molecule of species S_i.
v_T	Total number or vinyl groups in simulation. Defined as $\sum_{i=1}^{K} c_i v_i$
vol_i	Interaction volume of a molecule of species S_i
vol_T	Simulation volume.
k^{exp}	Experimentally obtained kinetic rate constant for propagation reaction between radical groups and vinyl groups.
$[L_i]$	Local concentration of vinyl groups for species S_i. Defined as $v_i vol_i^{-1}$.
$[G_i]$	Global concentration of vinyl groups for species S_i, which excludes the local vinyl groups. Defined as $(v_T - v_i)vol_T^{-1}$.
$f_{Z,i}$	Success rate of reaction between a radical in position Z with a vinyl of species S_i. $f_{Z,S_i} \in \{1, \phi, \psi, \Omega\}$. The used values can be found in Table 4.1
$\mathcal{R}_n$	Reaction with identifier $n \in \{1, 2, ..., N\}$. Either a monomolecular or bimolecular reaction with a single reaction product.
k_n	Rate constant of reaction $\mathcal{R}_n$, given in $\mathrm{l\,mol^{-1}\,h^{-1}}$ for bimolecular reactions and in $\mathrm{h^{-1}}$ for monomolecular reactions.
R_n	Reaction rate of reaction $\mathcal{R}_n$, defined as $R_n = [S_i][S_j]k_n$ for intermolecular reactions between species S_i and S_j and as $R_n = [S_i]k_n$ for intramolecular reactions of species S_i.

Chapter 1

Introduction

This work present an efficient method to simulate crosslinked RAFT copolymerization, a polymerization process with which nanogels, which are a type of nanocarrier, can be created. In this chapter we will first explain the importance of nanocarriers in drug delivery. We will highlight the effects of size and structure on the usage of nanogels as nanocarriers. Subsequently, in Section 1.2 we will explain the polymerization process by which these nanogels are created. In Section 1.3 we will explain the need for simulations of chemical processes and list different simulation methods. After that, in Section 1.4 we define the requirements which our simulation of the polymerization process of nanogels should fulfill. Section 1.5 lists the related work done in this field. It gives an overview of chemical models and simulations of crosslinker polymerization systems. Finally, in Section 1.6, we give an outline of the structure of this report.

1.1 Motivation

Nanomedicine is an emerging field in which nanotechnology is used to detect and treat diseases. A recent trend in nanomedicine is the use of nanocarriers: nanoparticles that are used as carriers for drugs and imaging agents. Imaging agents are molecules that help increase contrast in medical imaging (e.g. MRI scans). By manipulating the size, shape and surface of nanoparticles, it is possible to target specific tissues [3,4].

Targeted drug delivery is especially promising in the field of cancer treatment. Small-molecule therapeutics that are currently in use come with significant side effects due to the interaction with both healthy and unhealthy cells, such as hair loss, nausea, vomiting and immunosupression. By encapsulating these small-molecule therapeutics and only releasing them at the site of interest, using nanocarriers, interaction with healthy cells can be greatly diminished [3].

An effective carrier has three key properties:

1. avoiding filtering from the blood for a sufficient amount of time
2. accumulating in the site of interest and
3. releasing the drugs or imaging agent in the unhealthy cells for treatment.

We will now explain the effects of nanoparticle size on the first two properties and show that there is a trade-off to be made when choosing a nanoparticle size.

1.1.1 Size effect on blood filtering

Avoiding filtering from the blood is important for the targeting, because the longer a particle remains in circulation, the higher the probability of it entering the tumor tissue. The size of the particles is one of the major ways to prevent filtering from the blood. Particles smaller than 6 nanometers would be filtered out of the bloodstream by the kidneys and particles larger than 150 nanometers would be filtered out by the spleen and liver. This makes nanoparticles between 10 and 100 nanometers in size the most suitable sized carriers. Within this size range, the larger nanoparticles are generally cleared slower from the bloodstream, giving the particles more time to accumulate in tumor tissue [4,5].

1.1.2 Size effect on particle accumulation

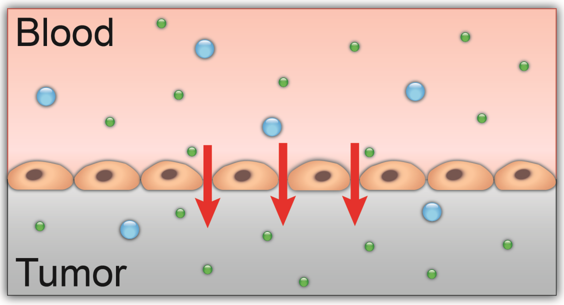
Particle size also has an influence on the site of particle accumulation. Smaller particles, usually around 10-30 nanometers, can most easily move through the arterial wall to leave the bloodstream in a process called extravasation and deeply penetrate the tumor tissue. In contrast, larger particles around 70-100 nanometers, undergo less extravasation and accumulate nearer to the blood vessels, often in the space between cells called the extracellular matrix (ECM) [4,5,6,7]. This is partly caused by the uptake mechanism by which cells are entered. By interacting with sufficient receptors on the cell membrane, the uptake mechanism is triggered in which the cell membrane wraps around the molecules that are interacting with the receptors, thus bringing it into the cell. If particles are large, they interact with many receptors, possibly hindering uptake by creating a local depletion of receptors. This leads to larger particles being enveloped in vesicles by themselves while leaving a less responsive cell membrane, leading to reduced uptake. If particles are small, multiple particles may be needed to trigger the uptake mechanism, which may also decrease uptake. The optimal particle size range for cell uptake is between 25 and 50 nanometers [4,8]. Both the extravasation and uptake mechanisms are illustrated by Figure 1.1.

1.1.3 Drug release

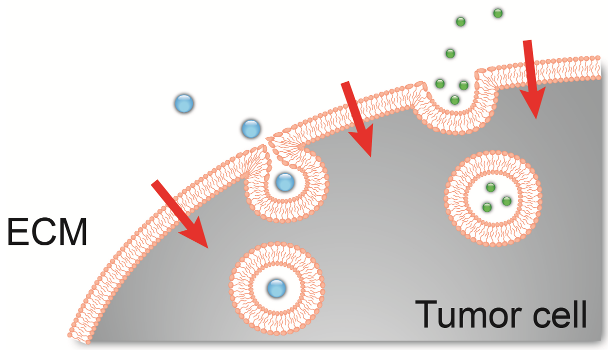
The third key property for nanocarriers, drug release, can be fulfilled by using either internal or external stimuli [4,9]. External stimuli, such as near-infrared light inside the target tissue, are limited to surface tissues as it is hard to reach deeper organs. Typical internal stimuli are differences between the tumor environment and the environment of healthy tissue. The tumor environment has a low pH, low oxygen concentration or a higher concentration of specific enzymes. Once the needed stimulus is delivered, the nanoparticle will detach from the drug molecules or break down to release it.

1.1.4 Optimal particle size

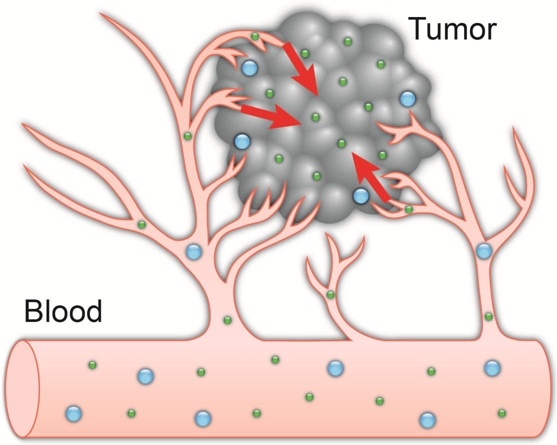
Optimal particle size depends on the application of the nanocarrier. If the nanocarrier is used for medical imaging, it only needs to stay in the body until the imaging process is finished. As such, relatively rapid clearance from the body is strongly preferred to keep side-effects to a minimum. This is most easily achieved by the filtering of the blood by the kidneys, which requires small particles [10]. However, the blood half life, i.e. the time before half the particles are removed from the blood, should still be sufficiently large for the nanogel to be able to leave the bloodstream and enter the tumor tissue. In contrast, if the nanocarrier is used for drug delivery it should remain in the target tissue for longer, so there is enough time to release the drugs This requires large particles. For optimal drug delivery this drug should penetrate the tumor thoroughly, which in turn requires using smaller particles. It is thus clear that the optimal particle size must be chosen by striking a balance between



(a) Particle size is crucial for extravasation, especially near tumors. Smaller particles can more easily move through the arterial wall to leave the bloodstream.



(b) Uptake mechanism and rate are influenced by particle size. Large particles can induce the membrane wrapping mechanism of a cell by itself, whereas multiple small particles are needed. Larger particles will thus get their own vesicle, leading to ineffecient uptake.



(c) As an effect of both more uptake and more extravastation, smaller particles penetrate deeper into tumors than large particles, which stay closer to the blood vessels.

Figure 1.1: Particle size influences tissue targeting [By M. Rachèl Elzes].

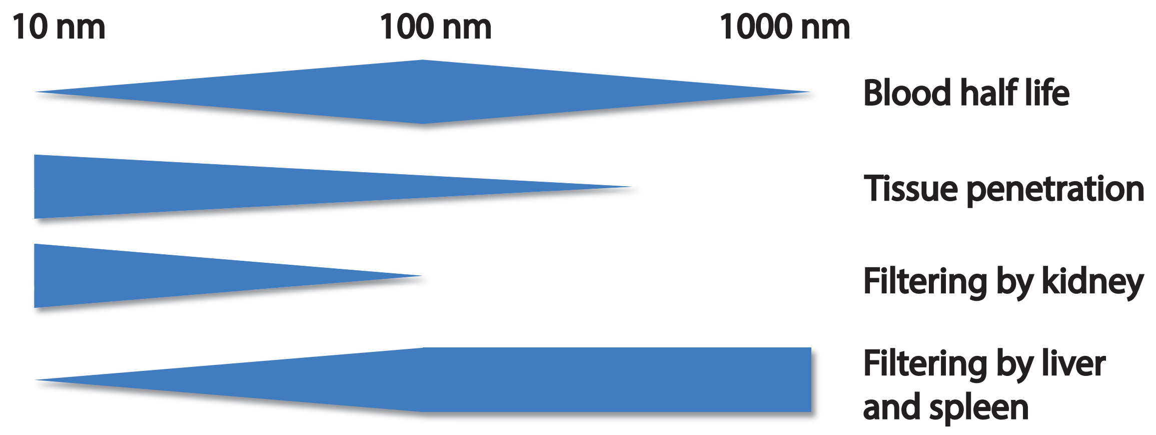


Figure 1.2: General trend of size effects of nanoparticles. A thinner bar indicates a less prominent effect. Recreated from [10]

blood filtering, blood half life and tumor penetration. A short overview of these important effects is given by Figure 1.2.

The dispersion of the particles, i.e. the variation of the particle sizes, also plays an important role. By using nanocarriers of multiple sizes, the advantages of larger and smaller particles can be combined. However, evaluation of the impact of nanocarriers with a high polydispersity, i.e. a wide range of particle sizes, is complex and a high polydispersity is thus not recommended [11,12].

1.1.5 Nanogels

One possible type of nanocarrier is the type central to this work, namely a nanogel: a hydrophilic nanoparticle with a network structure. This network structure is made out of interlinked polymer chains, chains consisting of repeated subunits called monomers. By filling the pores in the network structure with other molecules, the nanogel can act as a transport vessel for these molecules. Another property of nanogels is activated swelling, whereby a specific trigger, e.g temperature or pH, may activate the absorption of large amounts of solvent by a nanogel; this enlarges the particle, and thus the pores, releasing the transported molecules from them [13,14]. It is thus not only important for a nanogel to have the right size, but it should also have large enough pores to be able to be used as a nanocarrier.

1.2 Nanogel preparation

Nanogels consist of polymer networks, as mentioned previously. These polymer networks can be created via the process of crosslinked controlled radical polymerization, which will be further explained in this section. We will first give a general outline of the polymerization process. Subsequently, we will explain the 'radical', the 'controlled' and the 'crosslinked' aspects of crosslinked controlled radical polymerization.

1.2.1 Polymerization

Polymerization is the process of creating polymers by chaining repeated subunits called monomers together to form chains, often by only a single reaction mechanism. A typical polymerization process consists of three steps: initiation, propagation and termination.

In the initiation step, at the start of the polymerization process, the initial reactive chain ends are created. In the second step, propagation, these chain-ends react with monomers, creating a new reactive chain end, allowing the propagation reaction to continue. In the last step, termination, the growth of the polymer chain is stopped by a non-reversible reaction that deactivates the reactive chain end.

When termination happens in parallel to propagation, some polymers stop growing whereas other can continue, yielding polymers of many different sizes and thus a higher polydispersity. In that case termination is considered an unwanted side reaction, as one optimally would want a low polydispersity, as previously mentioned in Section 1.1.4. Deliberate termination can be induced by the (manual) addition of a terminating agent.

1.2.2 Radical polymerization

There are many types of polymerization reactions, but the one we focus on in this research is Free Radical Polymerization (FRP). Radicals are molecules with a reactive group containing an unpaired electron, also known as a free radical electron. Electrons are only stable in pairs; this makes radicals extremely reactive. The creation of polymers via radical polymerization follows the usual steps: initiation, propagation and termination.

Initiation Initiation for radical polymerization starts with the creation of the radicals. Because radicals are extremely reactive they are often created *in situ* by splitting a so-called initiator, forming two initiator radicals, as shown in Figure 1.3. We will indicate radicals using a black dot representing the unpaired electron. For clarity we will refer to the split halves of the initiator as **half-initiators** once they are part of a polymer, at which point they are not radical anymore. The initiators, initiator radicals and half-initiators are indicated in blue.

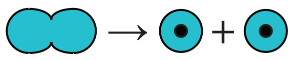


Figure 1.3: The initiator is broken into two initiator radicals. Radical molecules are indicated with a black dot.

Subsequently, this newly created radical can react with vinyl molecules, which are molecules with a vinyl group. A vinyl group is, by definition, a reactive group containing two double bonded carbons and three hydrogen atoms, as illustrated in Figure 1.4. A radical-vinyl reaction connects the radical and vinyl molecules and yields a new radical electron placed on the vinyl molecule, as shown by Figure 1.5. By this initial radical-vinyl reaction the initial polymer chain is created. The reactive center on the polymer will be indicated using orange. Typical initiators for radical polymerization are thermal initiators, which decompose into radicals with the application of heat, allowing the usage of external triggers for the polymerization process [15,16].

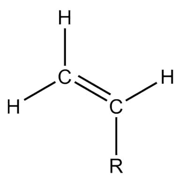


Figure 1.4: Structure of a vinyl group. A vinyl group contains two double bonded carbon atoms (C), three hydrogen atoms (H) and the rest of the molecule (R).

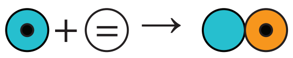


Figure 1.5: Initiator radicals (blue) react with vinyl monomers (white) to form the start of a polymer chain. The growth of the polymer chain can continue from the reactive center (orange). Vinyl groups are indicated by two parallel lines, representing the double bond in the vinyl group, as shown in Figure 1.4.

Propagation The radical-vinyl reaction between the reactive radical and vinyl groups, yielding a new radical group, is what drives the polymerization process in FRP. The growth of the polymer can continue with more radical-vinyl reactions between the new polymer chain and monomers. This way linear polymer chains form with a radical electron at one end and the half-initiator at the other, as illustrated by Figure 1.6. This chain growth is also referred to as chain propagation and the side at which the growth occurs is called the reactive center of the polymer.

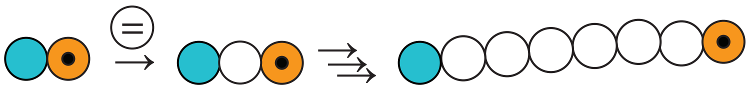


Figure 1.6: The start of the polymer chain reacts with multiple vinyl monomers (white) to form a linear polymer chain with the half-initiator at one end (blue) and a radical monomer at the other (orange).

Termination Other possible reactions are the ones between radicals, which yield unreactive molecules. Since radical-radical reactions thus terminate the polymerization process they are considered unwanted side reactions, as explained in Section 1.2.1.

1.2.3 Controlled radical polymerization

To reduce the unwanted effects of termination, Controlled Radical Polymerization (CRP) was invented, which is a polymerization process in which termination is greatly diminished. The specific controlled radical polymerization process we focus on in this work is called Reversible Addition-Fragmentation chain-Transfer (RAFT) polymerization.

In RAFT polymerization a molecule called a RAFT agent undergoes *reversible* reactions in which it temporarily bonds itself with free radicals to slow down the polymerization process [17,18]. The RAFT agent attaches to the reactive end of the polymer, the radical electron is transferred to the RAFT agent, where it is not reactive. This blocks the chain from propagating. The RAFT agent is not considered to be part of the polymer itself, since the bond that is formed between RAFT agent and polymer is semi-stable. This means that the polymer chains do not become dead chains, but rather temporary *inactive chains*.

A RAFT agent generally has two binding sites with which it can be attached to these reactive centers, but only one binding site can be detached at a time. As such, RAFT agents are introduced to the reaction mixture with typically one half-initiator connected to one of the binding sites. The **initial equilibrium** will have RAFT agents that are still attached to these half-initiator they start with. This chemical reaction for the initial RAFT equilibrium is illustrated in Figure 1.7. Once polymers are created, the RAFT agents will attach themselves to and detach themselves from polymers instead of half-initiators. We refer to this as the **main equilibrium**, which is shown in Figure 1.8.

The initiator serves an additional purpose in CRP. In FRP the initiator serves as the primary source of initiator radicals, whereas in CRP the initiator is also used to induce the release of the half-

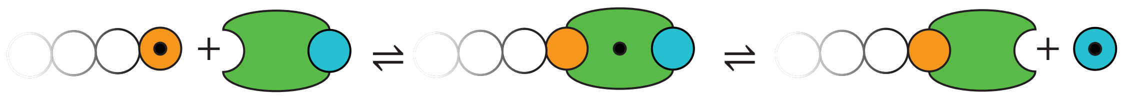


Figure 1.7: Initial RAFT equilibrium. The RAFT agent (green) undergoes an addition reaction with a radical polymer to form a two-arm adduct (middle). When this undergoes fragmentation it either yields the polymer or the half-initiator (blue). Note that the reactive center of the polymer becomes inactive when attached to the RAFT agent, as the radical group is removed.

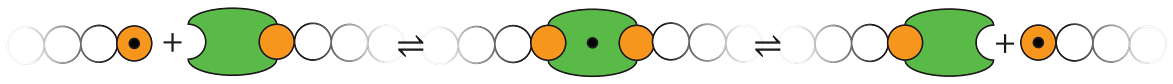


Figure 1.8: Main RAFT equilibrium. The RAFT agent (green) attaches itself to the reactive center (orange) of the polymer chain with addition reactions, forming a two-arm adduct which can then be broken up again by a fragmentation reaction which releases either of the polymers.

initiators connected to the RAFT agents via the initial RAFT equilibrium. In this work we consider two cases. Either 1) the initiator splits slowly and the initiator radicals it contributes are negligible, with the primary purpose of the initiator being the trigger for release of half-initiators via the RAFT equilibrium or 2) the initiator splits near-instantaneously and is fully converted to initiator radicals at the start of the simulation, while the created radicals also serve as the trigger for half-initiator release via the RAFT equilibrium.

The RAFT mechanism has several effects on the polymerization process:

- The propagation rate is lower due to the lower number of active reactive centers. The polymerization process in presence of RAFT agents can take hours where it would take seconds without such agents.
- A sharp decrease in termination due to the lowered chance of two radicals encountering each other.
- A decrease in polydispersity as an effect of both less termination and a lower propagation rate, leading to polymers that can keep on propagating. This effect is illustrated in Figure 1.9.
- Introduction of half-initiators with reactive centers via RAFT agents.

1.2.4 Crosslinked polymerization

It is also possible to create more complex shapes than linear chains by introducing molecules with multiple reactive groups to the reaction mixture. It is possible to create many different shapes of polymers depending on the number of reactive groups on the monomers: e.g, ring-shaped polymers, star-shaped polymers or comb-shaped polymers [20]. Network polymers can be created by including crosslinkers, which are molecules with a vinyl group on both ends, which have the ability to link two polymer chains together. This process is known as crosslinked polymerization and is a specific type of branched polymerization. The connection formed by a crosslinker is called a crosslink

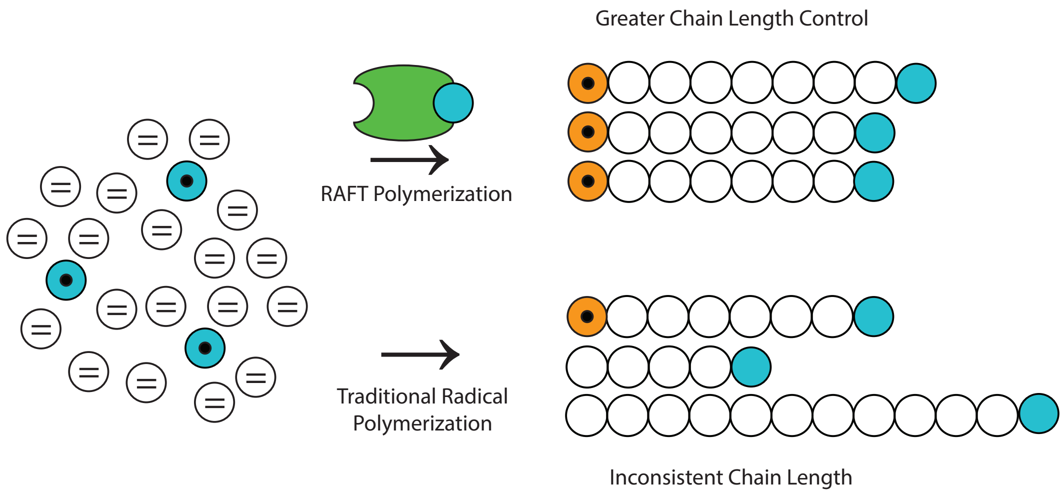

Greater Chain Length Control
RAFT Polymerization
Traditional Radical Polymerization
Inconsistent Chain Length

Figure 1.9: General comparison of polymers made with traditional radical polymerization against those made using RAFT process. Traditional radical polymerization also yields unreactive polymer chains due to termination. Recreated from [19].

and it introduces a branching point on both polymer chains. The difference between linear, (non-crosslinked) branched and crosslinked polymers is visualized in Figure 1.10. There are two main types of crosslinking [21]:

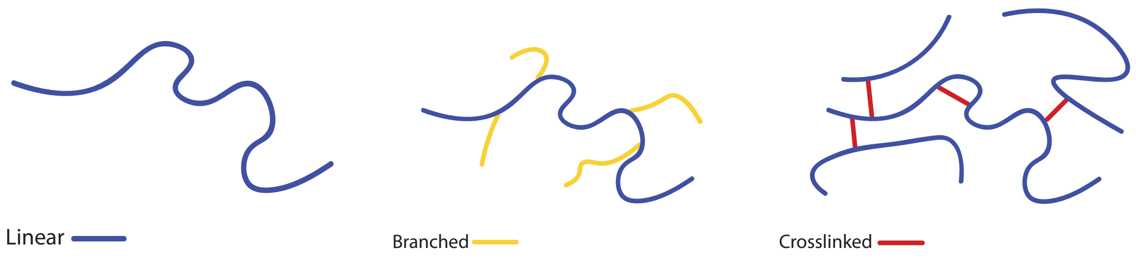


Figure 1.10: Polymer types

- **Post-polymerization crosslinking**: After multiple polymer chains have been created, a crosslinker is added which binds the already created chains together.
- **Copolymerization crosslinking (CC)**: The crosslinkers are connected to the polymer chains during the polymerization process. In this case the crosslinker is thus a part of the chain(s) and not something that is later attached. Such crosslinkers are preferably similar to the monomers used, in terms of solubility and reactivity. The main difference with the monomer is that a crosslinker is reactive on two sides. Propagation can thus happen on both sides of the molecule.

The specific crosslinker we focus on in this work is a difunctional monomer: a monomer with a reactive (vinyl) group on both sides. Since propagation happens with both monofunctional and difunctional monomers the resulting process is a copolymerization. A schematic representation of propagation in copolymerization is shown in Figure 1.11.

Polymers with both radical and vinyl groups can also react with themselves, a reaction we refer to as intramolecular crosslinking or cyclization, as polymer cycles are create by this reaction. Flory-Stockmayer Theory, which is the basis for many crosslinked polymerization models, assumes that

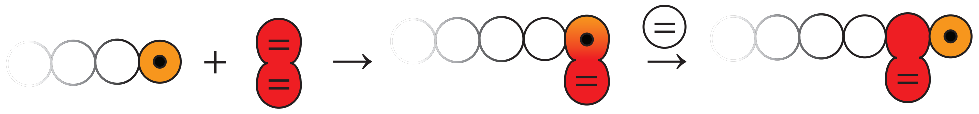
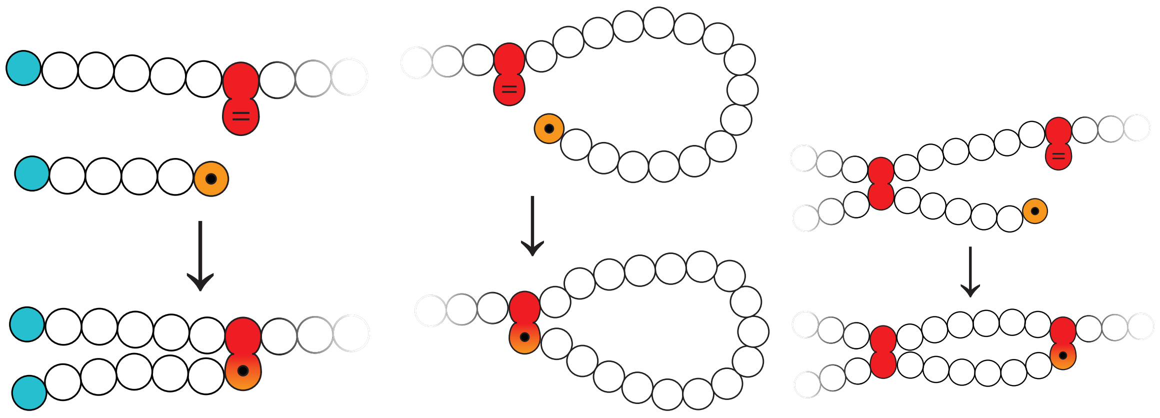


Figure 1.11: Copolymerization reaction with (monofunctional) monomers (white) and crosslinkers (red). Vinyl groups are introduced to the polymers by propagation reactions with crosslinkers, which can undergo further radical-vinyl reactions.

the amount of cyclization is negligible [22,23]. However, this has been proven incorrect for many cases; intramolecular reactions have a significant effect on the overall polymerization at higher ratios of crosslinker to monomer and at low concentrations of monomers [2,24,25].

A schematic overview of both intermolecular and intramolecular crosslinking is given by Figure 1.12 and an overview for the overall crosslinked coplymerization is given by Figure 1.13.



(a) Intermolecular crosslinking *(b)* Intramolecular crosslinking: Primary cyclization *(c)* Intramolecular crosslinking: Secondary cyclization

Figure 1.12: Types of crosslinking reactions

When a polymer network becomes big enough, the polymers become less soluble and a gel-like substance forms. This is also known as gelation, or reaching the gel point. At the gel point many of the polymers interconnect, forming large networks. This interconnection can continue until there is only a single solid polymer network left. Since nanogels need to be rather small, the polymerization process is stopped before gelation occurs. The overall polymerization reaction, including gel point, is illustrated by Figure 1.14.

Crosslinked CRP CRP has an important effect on crosslinked copolymerization. With traditional free radical polymerization the size and composition of the emerging polymers is wholly dependent on the local concentration of monomers around the reactive centers of the polymers. Because the reactive center remains active, it easily reacts with everything in its immediate vicinity, including its own reactive groups, both radical groups and vinyl groups. This causes dense clusters to be formed, containing few reactive groups.

CRP reduces localized effects, which in turn leads to a more homogeneous rate of branching throughout the polymer networks. The differences in rates of branching between FRP and CRP have been visualized in Figure 1.14. The rate of branching is important for the intended use of the polymer; densely connected polymers do not have the same pores that make the polymers suitable for their use as nanocarriers. Note that CRP has less effect on polydispersity in a crosslinked

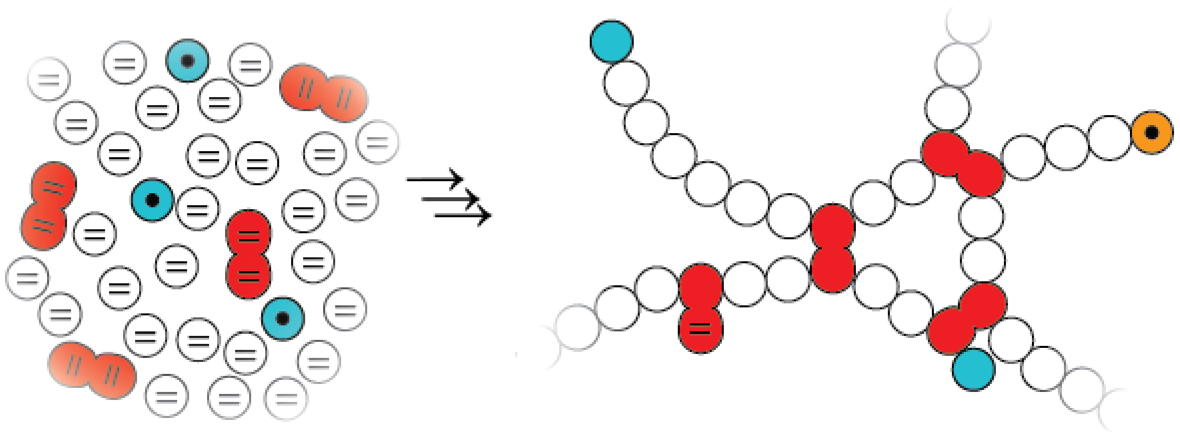


Figure 1.13: Schematic overview of crosslinked copolymerization.

copolymerization and is primarily used to prevent these localized effects.

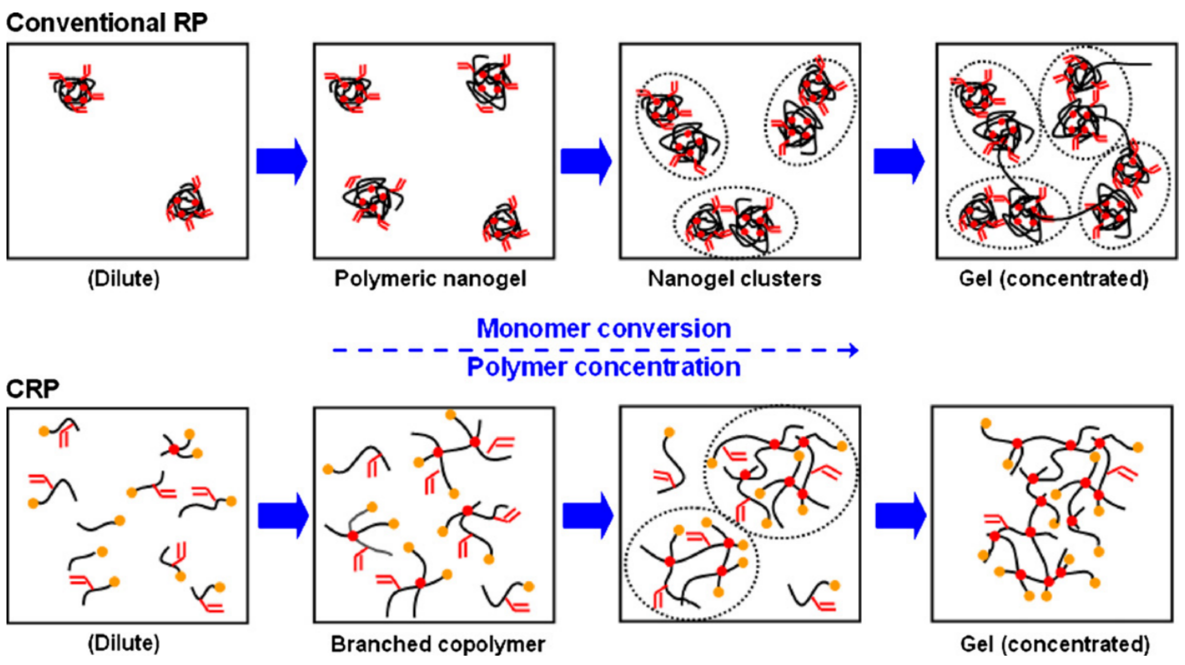
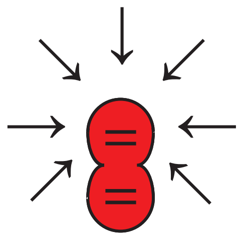


Figure 1.14: Different gelation processes between (F)RP and CRP [26]. The red lines represent vinyl groups and the orange circles reactive centers.

1.2.5 Steric hindrance

Steric hindrance occurs when accessibility of reactive sites is reduced by parts of a molecule getting in the way, preventing otherwise possible reactions [21,27]. This effect also has to be taken into consideration in crosslinked copolymerization. Whereas a crosslinker that has not reacted yet can be attacked by radicals from each angle, a pendent vinyl group can be approached from fewer angles. As a result, fewer reactions occur with pendent vinyl groups than the vinyl groups of unreacted crosslinkers. This can have a significant effect on the propagation rate with pendent vinyl groups [28,29]. The steric hindrance effect is illustrated in Figure 1.15 and will be further discussed in

Section 4.2.


(a) Schematic representation of possible attack angles of a crosslinker that has not yet reacted.

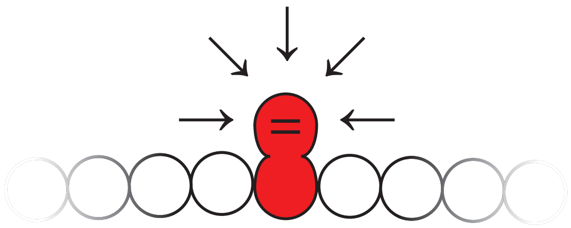

(b) Schematic representation of possible attack angles of a pendent vinyl group in a polymer.

Figure 1.15: A pendent vinyl group in a polymer can be attacked from fewer angles than a crosslinker that has not yet reacted.

1.3 Simulation

The process by which nanogels are created, is relatively new and as such many experiments may be needed to obtain the polymers with the wanted properties. These experiments are both resource and time consuming, making this an expensive undertaking. Simulations of the chemical process can be used to gain more insight into the polymerization process and reduce the number of experiments needed to obtain polymers with the sought-after properties.

There is a plenitude of methods to simulate chemical reactions, with varying abstraction levels. The most detailed simulations include full 3D models of the molecules, molecule location, velocity and direction. They can predict whether a collision or a reaction occurs when 3D models collide, by checking whether there is enough velocity to initiate a reaction. This type of simulation is known as Molecular Dynamics (MD) [30,31,32].

A less detailed version of this method is the Dynamic Lattice Liquid (DLL) method, where the movement of the molecular models is limited to a 2D or 3D grid [31,33]. The usage of a grid greatly reduces the difficulty of calculating collisions.

Simulation methods that do not include molecule locations also exist. For homogeneous mixtures, like we have in a controlled polymerization, it is possible to accurately simulate reactions in the mixture without tracking particle locations. Concentrations can be used to calculate the average number of collisions, and the number of reactions that occur due to a collision depends on the activation energy threshold for the reaction. This means that the number of reactions can be accurately predicted using a kinetic scheme, which describes the reactions that can occur between molecules and at what rate this happens. A typical reaction rule in the kinetic scheme for a reaction between molecules A and B by which molecule C is created looks like

$$A + B \xrightarrow{k_{AB}} C \tag{1.1}$$

with kinetic rate constant k_{AB} that describes the speed at which this reaction occurs, which is dependent on likeliness of a reaction occurring when a collision happens. Using this kinetic rate constant we can calculate the reaction rate of this reaction with

$$R_{AB} = [A][B]k_{AB} \tag{1.2}$$

where $[A]$ and $[B]$ are the concentrations of molecules A and B respectively. Because a kinetic scheme gives the reaction rate of every possible reaction, it can be used to describe the shift in overall concentrations in the reaction vessel.

1.3.1 Deterministic simulation

A popular approach to kinetic simulations is using Ordinary Differential Equations (ODE) [31,34,35]. By converting the reaction rules of the kinetic scheme to describe the shifts in concentrations we get differential equations. One equation is created for each molecule species, and each equation has a term for every reaction in which the associated molecule species is either part of the reactants or product. For example, Equation 1.1 can be converted into the differential equations

$$\frac{d[A]}{dt} = -[A][B]k_{AB}, \ \frac{d[B]}{dt} = -[A][B]k_{AB} \text{ and } \frac{d[C]}{dt} = [A][B]k_{AB}, \tag{1.3}$$

which describe the shifts in concentration for reactants A and B and product C. The advantage of this approach is that it gives the same answer every time, making this a deterministic process. The speed of such a simulation will also be independent of the initial concentration of molecules. A major disadvantage is that there has to be an ODE for each molecule species in the simulation. This means that usually all possible species and reactions have to be known at the start of the simulation. This makes ODE simulation an inefficient approach to simulating chemical systems in which an arbitrarily large number of different molecules can be created. This is also the case when simulating crosslinked polymerization reactions, where most molecules contain both vinyl groups and radical electrons, meaning that most molecules can react with each other, each combination of molecules having a unique associated reaction rule. The number of reactions thus grows quadratically with the number of molecule species.

1.3.2 Stochastic simulation

Processes that are too complex to simulate using deterministic approaches are usually approximated by using stochastic techniques. The most widely used simulation technique for kinetic stochastic simulation was introduced by Gillespie and is often referred to as Kinetic Monte Carlo (KMC) simulation [36]. Instead of tracking all possible reactions that can occur between all possible molecules that can be created, a KMC simulation only tracks a limited number of molecules in the simulation and tracks the reaction rates of the reactions that can happen between those tracked molecules. This number is far smaller than all possible reaction rates.

At every step of the simulation a random reaction is picked using Monte Carlo techniques, after which the picked reaction will be simulated and both the tracked molecules and the reaction rates are updated. Because this system is stochastic, every simulation will yield different results, but the results obtained by a deterministic simulation can be approximated by using a large enough number of simulated molecules. The major advantages of the stochastic approach are the increase in speed by only tracking a subset of possible reactions, and the more dynamic nature of the simulation, which allows for easier tracking of molecule properties. A small disadvantage is a loss in accuracy, that can somewhat be negated by increasing the resolution of the simulation. Another disadvantage is that even with a subset of the possible reactions, when simulating a chemical system in which an arbitrarily large number of possible molecules can be created, the number of possible reactions will grow quadratically.

1.4 Research goals

Our main goal is to gain more insight into the crosslinked RAFT copolymerization process by simulating it. We would like to be able to predict the effects of changing reaction parameters. Whereas changing a single parameter per experiment in the lab is a long and arduous process, it can be done with the click of a button in a simulation. To fulfill our goal we aim to create a simulator which fulfills the following requirements:

R 1 **Inclusion of intramolecular reactions and steric hindrance** The chemical model that lies at the center of our simulation should include the effects of both steric hindrance and intramolecular reactions, as both have non-negligible influence on the polymerization process. This allows us to more accurately simulate highly crosslinked polymerizations at low concentrations.

R 2 **Dynamic input** Simulation parameters should be changeable by the user. Changeable parameters should at least include:

- Initial concentrations of initiator radicals, monomers, crosslinkers and RAFT agents.
- Weight of initiator radical, monomer and crosslinker molecules.
- Degree of steric hindrance.
- Initial number of molecules in the simulation.

R 3 **Dynamic output** The user should be able to select the data that will be extracted from the simulation. Possible options that should be included are:

- Average polymer size.
- A full Molecular Weight Distribution.
- Dispersity index, often referred to as Polydispersity Index (PDI), which is a measure of the distribution of molecular masses [37].
- Crosslink density, used for describing the openness of the polymer networks.
- A 3D model of a polymer in the simulation. This is used for visual feedback on the simulation process.

R 4 **Efficient scaling** The simulation time should scale well with increasing numbers of simulated molecules. We aim for a worst-case time complexity that is better than quadratic in the number of initial molecules.

R 5 **Multiple data points** The simulation should be able to generate output data for multiple points of progression of the polymerization reaction. We have chosen to use conversion, which is the percentage of vinyl groups that has reacted, as a scale of progression.

We choose to focus on stochastic simulations, as the dynamic output of Requirement R 3 fits well with the dynamic nature of stochastic simulations. However, as mentioned before, traditional KMC is an inefficient method for simulating crosslinked copolymerization. As such, we have to find a suitable algorithm which *does* scale well with the number of molecules in the simulation, as stated by Requirement R 4.

The simulator will be validated by simulating the chemical experiments performed in [38] and comparing its results with the lab results obtained in that research, which show polymer size and PDI at different points of conversion for multiple reaction parameters. Our simulator should show similar trends as the ones observed in these lab results.

1.5 Related work

RAFT polymerization was pioneered by Chiefari in 1998. Shortly after, the first simulation of RAFT polymerization was done by Shipp and Matyjaszewski, where they compared their model of RAFT polymerization to models of other CRP techniques [39]. Since then extensive simulations have been done of Controlled Radical Polymerization (CRP). Al-Harthi gives a detailed overview of these simulations in [40].

Crosslinked Copolymerization (CC), however, is harder to simulate, because of the large number of possible reactions. Existing techniques do not scale well, making the simulation of such processes take a long time. Tripathi *et al.* have had success in simulating crosslinked polymerization by creating a hybrid simulation which simulates propagation reactions using ODE, and KMC for all other reactions [28,41]. The kinetic model proposed by Tripathi *et al.* includes steric hindrance, as the main goal of the research was to determine the extent to which monomers in a polymer chain negatively affect accessibility to nearby pendent vinyl groups. The hybrid technique used by Tripathi is also available in the commercial simulation tool PREDICI [42]. In [41] they showed that the hybrid algorithm was more than 2 orders of magnitude faster than the KMC algorithm in the simulation of linear polymerizations. This, however, is a fixed increase in speed. In Sections 1.3.1 and 1.3.2 we mentioned that the number of possible reactions in both ODE and KMC increases quadratically with the number of different molecule species. This makes ODE and KMC inefficient approaches for simulating crosslinked polymerization, as these traditionally track the number of possible reactions. As both these techniques were used to create the hybrid technique, we expect the resulting algorithm to share this problem, meaning that the hybrid algorithm will not be able to fulfill our scaling requirement.

Poly *et al.* were able to successfully simulate crosslinked CRP by tracking the reactive groups in polymers rather than the different polymers themselves [2]. They also include intramolecular reactions in their model and show the effects of rate constants, initial ratios and initial concentrations on gelation and polymer composition and structure. However, due to the use of their rather simplistic model they were unable to calculate the polydispersity or a molecular weight distribution, as this models only keeps track of averages of molecule characteristics, rather than separate values for each molecule.

Gao *et al.* used a DLL simulation of CC CRP to analyze the effect of concentration on gelation [26]. Their simulation showed that the gel point is postponed when lowering the number of initiator, monomer and crosslinker molecules in the simulation. This is in agreement with the results of lab experiments showing a postponed gel point when lowering the initiator, monomer and crosslinker concentrations. Polanowski *et al.* used a DLL simulation of CC CRP to analyze the effect of intramolecular reactions and different ratios and concentrations of initiators, monomers and crosslinkers on gelation [29]. The scale of both these simulations was limited by the use of the DLL technique. Only 1,000,000 molecules where simulated, including solvent molecules. As solvent molecules take up the bulk of the molecules in a mixture, this simulation only simulates the creation of few nanoparticles. As we aimed to simulate the creation of a large number of nanoparticles, using a number of monomers in the range of 1×10^8 to 1×10^9, this technique was deemed insufficient for our purposes.

An overview of this and further related work is given in Table 1.1.

1.6 Report organization

The remainder of this report is organized as follows. In Chapter 2 a more detailed background is given into the chemical process. Subsequently, in Chapter 3, the Kinetic Monte Carlo algorithm is

Table 1.1: Related research

Authors	CC	CRP	Research type	Ref. nr.
M.A. Al-Harthi	X	✓	Review	[40]
E. Pintos *et al.*	X	✓	Monte Carlo simulation	[43]
P. Ganjeh-Anzabi *et al.*	X	✓	Monte Carlo simulation	[44]
D. Drache *et al.*	X	✓	Monte Carlo simulation	[45]
A.K. Tripathi *et al.*	X	✓	Monte Carlo simulation	[41]
A. Feldermann *et al.*	X	✓	Monte Carlo simulation	[46]
O. Okay *et al.*	✓	X	Lab experiments	[47]
A.K. Tripathi *et al.*	✓	X	Monte Carlo simulation	[28]
S. Hamzehlou *et al.*	✓	X	Monte Carlo simulation	[48]
H. Gao *et al.*	✓	✓	Lab experiments	[26]
Y. Zheng *et al.*	✓	✓	Mathematical model	[49]
J. Poly *et al.*	✓	✓	ODE simulation	[2]
H. Gao *et al.*	✓	✓	DLL simulation	[50]
P. Polanowski *et al.*	✓	✓	DLL simulation	[29]

explained in detail. This is followed by the explanation of the chemical model that is used by our simulation, in Chapter 4. Then, in Chapter 5, we show the changes that were made to the Kinetic Monte Carlo algorithm to make it more efficient. Subsequently, in Chapter 6, we explain how the simulator was implemented around this simulation algorithm with a framework for dynamic input and output. In Chapter 7 we discuss our simulation parameters and the simulation results obtained by using these parameters. Finally, in Chapter 8, conclusions and recommendations are given.

Chapter 2

Chemical background

This chapter gives a description of chemical processes. First notations of reactions and molecular structures will be explained. Subsequently molecular structures will be explained in more detail, including different kind of chemical bonds and their associated 3D structures. Lastly, we will explain the mechanics of free radical polymerization in more detail, followed by an explanation of the crosslinked copolymerization and RAFT processes using the previously introduced structural notation.

2.0.1 Chemical notations

We will discuss the notations which are commonly used for chemical reactions. Let us take a look at the chemical equation for the electrolysis of water as an example:

$$2\,H_2O \;\rightarrow\; 2\,H_2 + O_2$$

The reactants are written on the left side of the arrow and the products which are created out of these reactants on the right side of the arrow. The law of conservation of mass states that the number of atoms must remain the same, which means that the number of atoms in the reactants has to balance the number of atoms in the product. Coefficients are used to balance out the chemical equation (e.g. the '2' before H_2O and H_2).

Chemical reactions can be classified into four categories [51]:

- **Combination/Addition reactions**: A + B $\rightarrow$ AB
- **Decomposition/Fragmentation reactions**: AB $\rightarrow$ A + B
- **Displacement reactions**: A + BC $\rightarrow$ AC + B
- **Exchange reactions**: AB + CD $\rightarrow$ AD + CB

It is also possible for reactions to only occur in the presence of a *catalyst*, a molecule that is needed in the reaction but is not consumed in the process (e.g. A + B + C $\rightarrow$ AB + C). Catalyst are usually written above the arrow: A + B $\xrightarrow{C}$ AB. However, this is not the only use of this notation. It is also possible to write reactants on top of the arrow to indicate the addition of a reactant. This can be especially useful when chaining reactions, i.e: A + B $\longrightarrow$ AB $\xrightarrow{B}$ ABB.

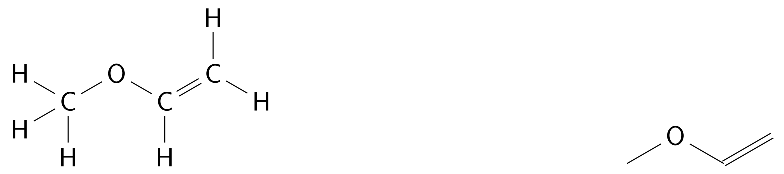
There are also other types of arrows which can be used:

- **Equilibrium arrows** are used to indicate an equilibrium reactions, which are reversible reactions, e.g. A + B $\rightleftharpoons$ AB

- **Triple arrows** are used to indicate a repeat of the previous reaction e.g.
  A + B $\longrightarrow$ AB $\xrightarrow{B}$ ABB $\rightrightarrows$ ABBBB$\cdots$

2.0.2 Molecular structures

Structure diagrams describe how atoms are connected to each other with bonds, but such diagrams may become confusing for larger molecules. To prevent this skeletal structure diagrams are used, which are a shorthand notation that can be used to prevent cluttered diagrams. Carbon atoms are indicated by line endings and meeting points of line segments, and hydrogen atoms are omitted altogether. The carbon atoms are connected to as many hydrogen atoms as are needed to fulfill the octet-rule, which will be explained in section 2.1.1. An example of a structural diagram and its corresponding skeletal diagram is shown in Figures 2.1a and 2.1b.



(a) Regular structure notation *(b)* Skeletal structure notation

Figure 2.1: Structural notations

2.1 Molecular aspects

Before we take a look at molecules as a whole, we will take a look at things on a smaller scale, namely atoms. Since the focus lies on organic chemistry, we will limit ourselves to hydrogen (H, atom number 1), carbon (C,6), nitrogen (N,7), oxygen (O,8), fluorine (F,9), phosphorus (P,15), sulfur (S,16) and chlorine (Cl,17). This section will first give an introduction to atoms and their electrons. Subsequently we will discuss the different kinds of chemical bonds and how they can be created by sharing or donating electrons. Then we will discuss atoms with too many or too few electrons and the effect on their reactivity.

2.1.1 Atomic aspects

An atom consists of a nucleus and electrons. The nucleus consists of positively charged protons and uncharged neutrons. The nucleus determines the type of the atom and always remains the same in regular chemical reactions. This type is often indicated by the atom number, which is the number of protons in the nucleus.

The electrons, which are negatively charged, are located in orbits around the nucleus, also known as **shells**. There are multiple shells, of which each one can hold a certain number of electrons. The inner shells are filled up first. The first three shell sizes, from inner to outer, are two, eight and eighteen. Using these sizes we can determine how many electrons an atom has in its outer shell.

For example: oxygen has an atomic number of 8, meaning that it has eight protons in its nucleus. For it to be neutrally charged it also needs to have eight electrons in its shells. Since the inner shells are filled up first we know that the inner shell must contain two electrons and the outer one six. Such electron configurations are represented by Bohr models, of which the one for oxygen can be seen in Figure 2.2.

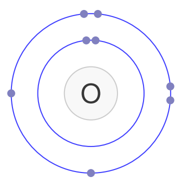


Figure 2.2: Bohr model of oxygen

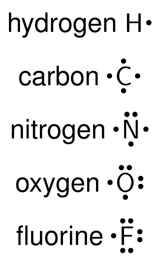


Figure 2.3: Example of Lewis Structures

The outer shell is the most important one, as it is the one which contains the electrons that are used in chemical bonds. This outer shell is called the **valence shell**, and the **valency** of an atom is the number of bonds it can have with other atoms. Note that a double bond counts as two and a triple bond as three towards valency. Since we can assume that the inner shells are completely filled, we can use a simplified notation, namely Lewis structures, also called electron-dot diagrams, which only show the electrons in the valence shell. Examples of Lewis structures can be seen in Figure 2.3.

Valence shells are most stable when they are either empty or full. Valence shells also tend to be stable when containing eight electrons, this phenomenon is referred to as the octet-rule [27]. As such, atoms will seek to fill their shells, which can be done by reacting with other atoms.

2.1.2 Chemical bonds

There are multiple types of chemical bonds. We will start by discussing the covalent bond and after that we will discuss the ionic bond. A **covalent bond** is created when two atoms each **share** an electron. Figure 2.4 shows some examples of covalent bonding. If, for example, a hydrogen atom shares an electron with another hydrogen atom, they will both have a fully filled valence shell, containing two electrons. Atoms can, depending on their valence electrons, also have multiple covalent bonds. The number of electrons shared between two atoms characterizes the type of bond. If two electrons (a single pair) are shared it is called a **single bond**, if four electrons are shared (two pairs) it is called **double bond** and if six electrons (three pairs) are shared it is called a **triple bond**, often indicated by a single, double or triple line respectively, as can be seen in Figure 2.4.

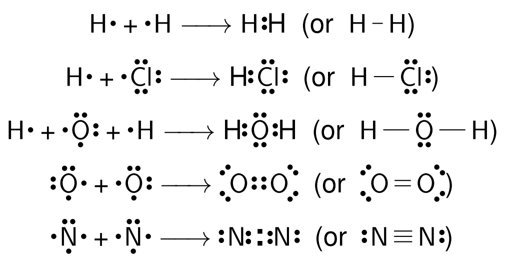


Figure 2.4: Lewis Structures of covalent bonding

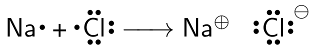


Figure 2.5: Lewis Structures of ionic bonding

An **ionic bond** is created when an atom **donates** an electron to another atom. An example can be seen in 2.5. Sodium donates its electron to chlorine, after which sodium has an empty valence shell and chlorine a full one. Sodium ends up with a positive charge, while chlorine ends up with a negative charge due to the difference between the number of protons and electrons that is caused by the donation. Due to this difference in charge they are attracted to each other, but can be easily pulled apart when putting it in a polar solvent. This is exactly what happens when dissolving (kitchen) salt in water.

2.1.3 Electronegativity

One thing that stands out is that NaCl is ionic while HCl is not, even though both hydrogen and sodium have only one electron in their valence shell. This is because the electronegativity, the ability of an atom to attract electrons, differs less between hydrogen and chlorine than between sodium and chlorine. The higher difference causes the sodium to *donate* the electron instead of just sharing it, while hydrogen and chlorine *share* electrons due to their similar electronegativity. Electronegativity can thus be seen as the pull an atom has on electrons.

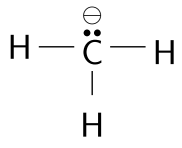
This also has another effect. A bond does not have to be ionic or covalent, but can actually lie somewhere in between (often called polar bonds), where an electron is shared but spends more time around one atom. This means that bonds lie on a spectrum and are not just one of two or three options [27]. We will not go into detail about this, as we will focus on polymerization, which focuses mainly on covalent bonds.

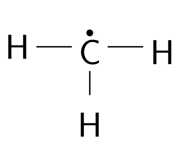
2.1.4 Ions, lone pairs and radicals

It is possible for an atom to have a valence shell that is not fully filled, even though atoms are most stable with a full outer shell. To show what effect this has we will take a look at methyl and its variations. The first example will be a methyl radical, $H_3C^{\bullet}$. We will then take a look at the methyl ions H_3C^{+} and H_3C^{-}.

A (free) **radical** is a molecule or atom with an unpaired valence electron, i.e. an electron that is not part of a bond or a lone pair. Radicals are highly reactive and are thus mostly created *in situ*. An unpaired electron is indicated by a single dot in molecule structures. When we consider the methyl radical in Figure 2.6b, we can see that there are a total of nine electrons and nine protons, which gives us a neutral charge. A radical is very reactive, as it tends to share its unpaired electron with other atoms to fill its valence shell. We will go into further detail about radicals and radical reactions in section 2.3. We will end this section with a short explanation of exceptional bonds and their representation.

Ions are molecules or atoms with a total number of electrons that is not equal to the total number of protons, resulting in a non-neutral charge. **Cations** are ions with a positive charge and **anions** are ions with a negative charge, indicated by $\ominus$. If we take a look at Figure 2.6c we can see that the carbon in a methyl cation only has six electrons in its valence shell, resulting in a positive charge, indicated by $\oplus$. The carbon thus has room for another shared pair of electrons. If we take a look at Figure 2.6a we can see that the carbon in a methyl anion has eight electrons in its shell, two of which are not shared. These unshared pairs are often called **lone pairs**, which can be indicated in molecular structures with two dots. *Lone pairs are usually not shown in structural formulae, unless involved in chemical reactions.* The carbon has a negative charge and will seek to share its lone pair.


(a) Methyl anion


(b) Methyl radical

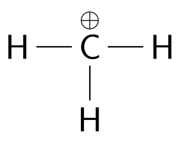

(c) Methyl cation

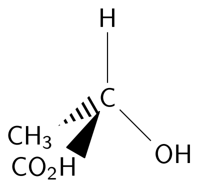
Figure 2.6: Methyl variants

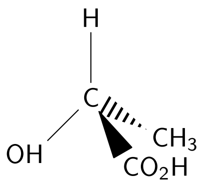
2.2 Molecular layout

In this section we will take a look at the 3D structures of molecules and how they affect chemical reactions. We will first take a look at bond orientation and bond rotation and how both affect molecule variety. Finally we will take a look at some examples where the 3D structure of a molecule influences the possible reactions which can occur.

2.2.1 Chirality

Molecules have certain aspects that can be included into our molecular model. One important aspect in chemistry is chirality, or handedness. Molecules that are not identical to their mirror images are chiral, which means that molecules with symmetry are thus achiral. The most common cause of chirality in organic chemistry are carbon atoms which are bonded with four different groups. Such carbon atoms are referred to as *chiral centers* [27]. Figure 2.7 shows an example of a chiral and achiral molecules. No matter which way the structure of the left-handed molecule is rotated, the structure of the right-handed variant cannot be created. Chirality is especially important when creating drugs, since our bodies might only be able to absorb the left-or right-handed variant of the drug. We will go into more detail about this in the following section.


(a) left-handed


(b) right-handed

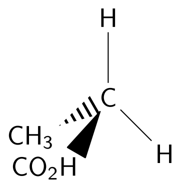
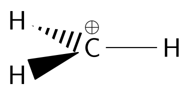
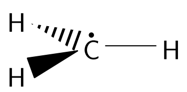

(c) achiral

Figure 2.7: Chirality examples


(a) Methyl cation


(b) Methyl radical

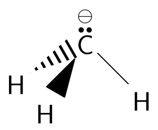
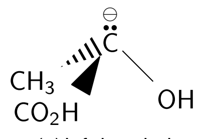
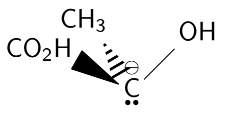

(c) Methyl anion

Figure 2.8: 3D structure of methyl variants

Another easily overlooked chiral center would be a carbon with three different bonded groups and a lone pair. The can be most easily shown by looking at Figure 2.8. While the 3D structure of the


(a) left-handed


(b) right-handed

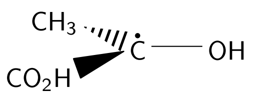

(c) achiral

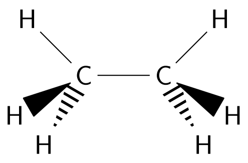
Figure 2.9: Chirality of carbon with lone pair

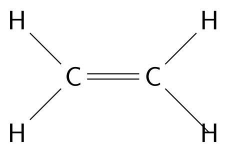
radical and cation are flat, the 3D structure of the anion is not. This means that if the molecule in Figure 2.9c would gain an extra electron in some way it could either become the molecule in Figure 2.9a or the one in Figure 2.9b, which are each mirrored versions of each other.

2.2.2 Bond orientation

Single, double and triple bonds have different structural properties. This is shown in Figure 2.10 using ethane, ethylene and ethyne. While a carbon with only single bonds has a tetrahedral structure, a carbon with a double bond is flat, as is one with a triple bond.

Because single bonds are free to rotate, molecules can adopt multiple conformations, as shown in Figure 2.11 [52]. We will treat such conformations as equal unless stated otherwise. Double and triple bonds cannot rotate. This has as effect that there are two possible orientations when a double bond is created. A chemical reaction can thus have multiple molecules with the same chemical formula, but with a different structure as a result. We call molecules with the same formula, but with a different structure **isomers**. We use the term **cis**-isomer for a double bond with two similar groups oriented in the same direction and use **trans**-isomer for a molecule that has a double bond with the similar groups pointed in opposite directions. An example of the cis- and trans-isomers of 2-butene is shown in Figure 2.12.


(a) ethane


(b) ethylene

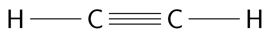

(c) ethyne

Figure 2.10: Bond angles

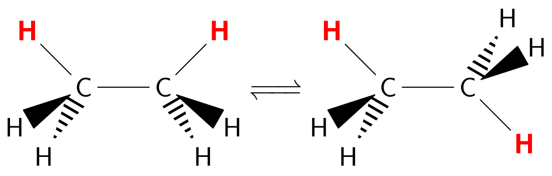


Figure 2.11: Bond rotation

2.3 Structural properties of radical polymerization

Previously, in chapter 1 we explained the three steps of radical polymerization, namely initiation, propagation and termination. In this section we will first describe the changes to molecule structures that happen at each of these steps in more detail. Subsequently, we do the same for the Radical Addition-Fragmentation chain-Transfer (RAFT) reactions. Lastly, we show propagation with both

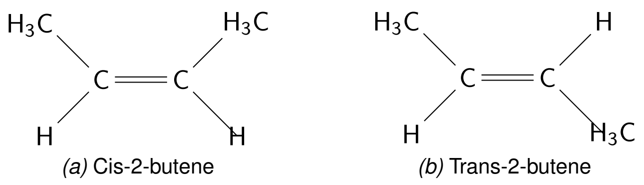


Figure 2.12: Isomers of 2-butene

monomers and crosslinkers.

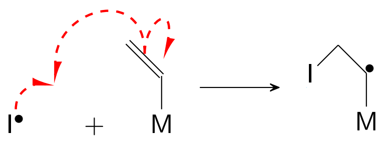


Figure 2.13: Electron rearrangement in radical addition reaction

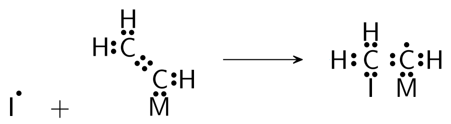


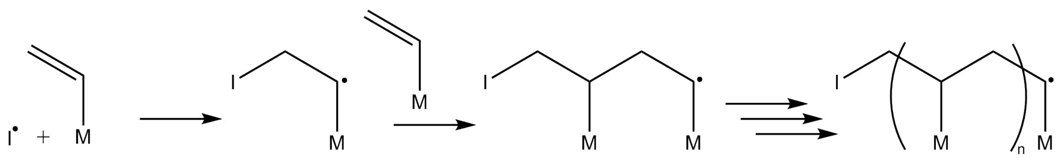
Figure 2.14: Lewis structure of reaction in Figure 2.13

- **Initiation**: Initiator radicals are created by splitting initiator molecules. The splitting of the initiator is relatively straightforward and is achieved by the splitting of the bond that connects the two halves of the initiator molecule, creating a free radical electron on each half. The following chemical equation represents the initiation reaction:

$$\text{I}-\text{I} \xrightarrow{\text{heat}} 2\,\text{I}\bullet$$

  The initiator radical is indicated with 'I•' and the dot represents the free radical electron. An example of the splitting of a specific initiator is given in Appendix A.

- **Propagation**: The radical will react with a monomer, connecting them and creating a free radical on the monomer. During the reaction the radical electron will react with a double bond, taking away one of the shared electrons of the double bond. This results in a bond between the radical and the double bonded carbon. The double bonded carbon has two single bonds after the reaction, one with the monomer and one with the (previously) radical molecule. A new unpaired electron is also formed, by the electron rearrangement shown in Figures 2.13 and 2.14. This new radical electron can then react with another monomer, thus propagating the radical addition reaction. This is the process which causes the polymer chain to grow. We will indicate a polymer chain consisting of n monomers with 'P$_n$'. A single monomer will be represented using 'M'. The following chemical equation represents the propagation reaction:



  The parenthesis around the monomer indicate a repeating pattern of n monomers.

- **Termination**: The polymerization can be stopped in multiple ways. The radical can, for example, react in such a way that no new radical is created. The most obvious example would be a radical-radical reaction, where two free radicals form a single bond, known as **recombination**. The following chemical equation represents the reaction for termination by recombination:

$$P_n^\bullet + P_m^\bullet \longrightarrow P_n — P_m$$

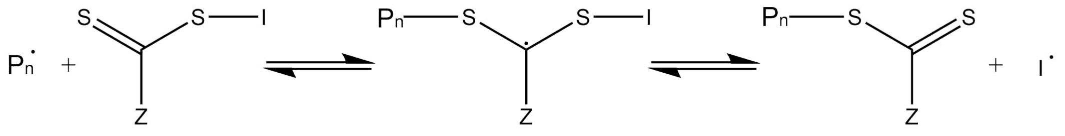
Another way of terminating is radical **disproportionation**, where instead of forming a bond, two radicals exchange a hydrogen atom for a free radical electron. The radical that receives the free radical electron forms a double bond at the place in the molecule where the exchanged hydrogen was located. After disproportionation the chains are not reactive anymore; such chains are referred to as **dead chains**. The following chemical equation represents the reaction for termination by disproportionation:

$$P_n^\bullet + P_m^\bullet \longrightarrow P_n + P_m$$

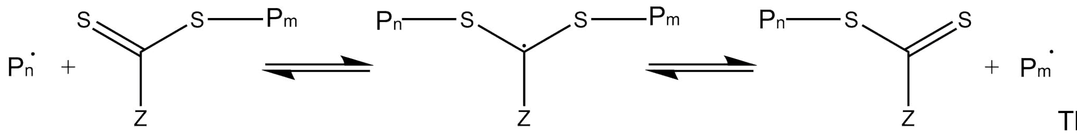
The chemical equation for disproportionation does not include the created double bond or the exchanged hydrogen atom, since the placement for those is decided by the specific molecules used. Instead, it simply indicates that the free radical electrons are not available anymore, thus resulting in dead chains. Specific examples for both recombination and disproportionation that *do* include structural details are given in Appendix A. It is important to differ between the two types of radical-radical termination, since they result in different molecule sizes, and thus influence the molecular weight distribution in different ways.

2.3.1 RAFT polymerization

The RAFT reaction has the following **initial equilibrium**:



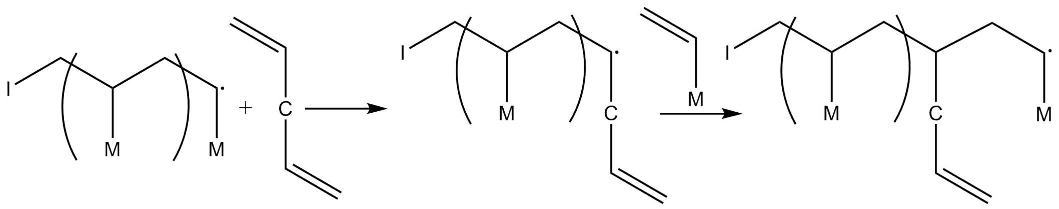
Here ‘S’ is a sulfur atom and ‘Z’ indicates a rest group which is not involved in the reaction. One thing to note here is that the initial equilibrium introduces a second initiator radical which can start propagation, which does not necessarily have the same structure as the one created from the initiator. More polymer chains will form after propagation has happened for a while, which leads to the following **main equilibrium** of the RAFT reaction:



The addition reaction of the RAFT agent is similar to the propagation reaction between a radical and a vinyl group, with the main difference being the reversibility of the RAFT reaction.

2.3.2 Crosslinked polymerization

The last reaction of which we will show the structural notation is the copolymerization reaction in which a pendent vinyl group is introduced. The following chemical equation shows this reaction.



Here 'C' is used to indicate a crosslinker, which is a symmetrical monomer with a vinyl group on both sides. Note that the crosslinker with a radical on one side and a vinyl group on the other usually is unable to react with itself, as the structure is not flexible enough to form a ring. Also note that both monomers and crosslinkers contribute similarly to the carbon backbone of the polymer chains.

Chapter 3

Simulation Background

This chapter serves as a description of existing Kinetic Monte Carlo (KMC) techniques. In Section 3.2 we discuss the KMC algorithm based on Gillespie's work [36,53]. In the subsequent section we describe an existing variation of this algorithm called Rejection Kinetic Monte Carlo (RKMC) [54].

3.1 Kinetic scheme

A KMC simulation has a kinetic model at its center which describes possible reactions and their rate constants. Say there are K molecule species S_i, $(i = 1, 2, ..., K)$ in the simulated volume vol_T, which can undergo N possible reactions $\mathcal{R}_n$, $(n = 1, 2, ..., N)$ with rate constant k_n, $(n = 1, 2, ..., N)$, given in $\mathrm{l\,mol^{-1}\,h^{-1}}$ for bimolecular reactions and in $\mathrm{h^{-1}}$ for monomolecular reactions. The rate constant is not actually constant but a function of temperature; however, we assume that the reactions occur at a constant temperature. In our case reactions are of the following types.

$\mathcal{R}_n := S_i + S_j \xrightarrow{k_n}$ reaction products (bimolecular reactions)
$\mathcal{R}_m := S_i \xrightarrow{k_m}$ reaction products (monomolecular reactions)

Each reaction has an associated reaction rate $\overline{\overline{R_n}}$, $(n = 1, 2, ..., N)$, measured in $\mathrm{mol\,l^{-1}\,h^{-1}}$. This reaction rate is a function of the concentration of the reactant molecules and describes the speed at which the reaction products are created. The reaction rate function for bimolecular reactions is

$$\overline{\overline{R_n}} = [S_i][S_j]k_n. \tag{3.1}$$

Here $[S_i]$ and $[S_j]$ are the concentrations in $\mathrm{mol\,l^{-1}}$ of molecule species S_i and S_j. For a monomolecular reaction the reaction rate function is

$$\overline{\overline{R_m}} = [S_i]k_m. \tag{3.2}$$

The reactions and their rate constants described by the kinetic model can be used to predict the shift in concentrations over time.

3.2 Kinetic Monte Carlo

A Kinetic Monte Carlo simulation uses a kinetic scheme to predict the shift in molecular concentrations using an algorithm consisting of the steps (1) initialization, (2) reaction selection, (3) time step calculation (4) reaction simulation, (5) iteration. We will now explain these steps in detail.

Step 1: Initialization The initial simulation state is created in the initialization step of the simulation, using the parameters provided by the user. Among these parameters is the kinetic model, containing the following properties:

- the K molecule species S_i, $(i = 1, 2, ..., K)$
- the N possible chemical reactions $\mathcal{R}_n$, $(n = 1, 2, ..., N)$
- the rate constants k_n, $(n = 1, 2, ..., N)$ of each reaction, at the simulated temperature
- the total number of initial molecules in the simulation c_T
- the initial concentration of molecules of each species $[S_i]$, $(i = 1, 2, ..., K)$

A simulation state consists of the number of molecules c_i of each molecule species S_i. We use the initial number of molecules in the simulation, c_T, to calculate the initial number of molecules of each species using

$$c_i := \frac{[S_i]}{\sum_{j=1}^{K}[S_j]} c_T. \tag{3.3}$$

Furthermore, we can use the concentrations and total number of molecules in combination with Avogadro's number, $A = 6.022 \cdot 10^{23}$, which is the number of molecules per mol, to define the total simulation volume, vol_T of the solution in liters:

$$vol_T := \frac{c_T}{A \sum_{i=1}^{K}[S_i]}. \tag{3.4}$$

It thus follows that the relation between concentration and number of molecules is

$$c_i = [S_i] A \ vol_T. \tag{3.5}$$

Since the number of molecules of each molecular species is tracked rather than the concentration, we also redefine the reaction rate accordingly. The reaction rate for a bimolecular reaction $\mathcal{R}_n$ between molecules of species S_i and S_j, *given in molecule* h^{-1}*, rather than* $\mathrm{mol\,l^{-1}\,h^{-1}}$, is

$$R_n = c_i c_j \frac{k_n}{A \ vol_T}. \tag{3.6}$$

and the reaction rate of a monomolecular reaction $\mathcal{R}_m$ of a molecule of species S_i is

$$R_m = c_i k_m. \tag{3.7}$$

As the last step in the initialization we set the reaction time (in hours) of our simulation, $t = 0$.

Step 2: Reaction selection Each iteration of the KMC algorithm starts with picking the reaction that is going to be simulated. The probability for reaction $\mathcal{R}_n$ to be chosen for the simulation is $R_n R_T^{-1}$ with

$$R_T = \sum_{m=1}^{N} R_m \tag{3.8}$$

Reaction selection starts with picking a uniformly distributed random number x in the interval $[0, 1)$ and subsequently selecting $\mathcal{R}_n$ for which the following inequalities hold

$$\sum_{m=1}^{n-1} R_m < x R_T < \sum_{m=1}^{n} R_m. \tag{3.9}$$

Step 3: Time step calculation After selecting the reaction, the reaction time t is updated to reflect the time that has passed between this reaction and the previous one. Following the Gillespie algorithm [36], upon which KMC is based, t is increased by

$$\Delta t := \frac{ln(y^{-1})}{R_T} \tag{3.10}$$

where y is another uniformly distributed random number in the interval $[0,1]$. This is equivalent to sampling an exponential distribution with a rate parameter $\lambda = R_T$ [43].

Step 4: Reaction simulation The simulation of the reaction consists of updating the number of the molecules of the reactants and products for the selected mono- or bimolecular reaction $\mathcal{R}_n$. For every S_i in the set of reactants in $\mathcal{R}_n$, c_i is decreased by one. For every S_j in the set of products in $\mathcal{R}_n$, c_j is increased by one.

Step 5: Iteration The reaction selection and reaction simulation steps are repeated till a halting criterion has been met. Common halting criteria are:

- There are no further possible reactions.
- A certain time has passed.
- A specific molecule species has completely reacted.

After the simulation is stopped the relevant data that is collected is combined to provide the wanted output formats. A typical output could be a molecular weight distribution.

3.3 Rejection Kinetic Monte Carlo

KMC simulations generally come in two types: rejection-free KMC, simply referred to as KMC, and rejection KMC (RKMC) [54,55]. The differences between these two types of KMC are the manner in which the reaction that is going to be simulated is chosen and how the time step is calculated.

Step 2: Reaction selection Whereas KMC uses separate reaction rates R_n for each reaction, RKMC does not. Rather than using the reaction rates to pick a random reaction, RKMC relies on sampling from a uniform distribution, meaning that each possible reaction has the same probability to be picked.

Reaction acceptation The reaction rates must be corrected for the error introduced by sampling from a uniform distribution. This is done by accepting or rejecting the chosen reaction, by introducing a success rate for each reaction $\mathcal{R}_n$. If a chosen reaction is rejected the RKMC algorithm continues from the reaction selection step again.

This success rate f_n must be chosen in such a way that $f_n = R_n R_0^{-1}$, where R_0 is a suitable upper bound, which must be large enough to guarantee $f_n < 1$ for all n. Performing the reaction acceptation is straightforward. A uniformly distributed random number z in the interval $[0,1)$ is generated and the reaction is accepted when $z < f_n$.

Step 3: Time step calculation Instead of using the sum of reaction rates, the time difference is calculated using

$$\Delta t = \frac{ln(y^{-1})}{NR_0} \tag{3.11}$$

where y again is a uniformly distributed random number in the interval $[0, 1)$.

The advantage of rejection KMC is that sampling can easily be done in constant time. Furthermore, the reaction rates do not have to be updated, as they only have to be calculated for the reaction that is selected. The disadvantage is that not every selected reaction is accepted, introducing extra overhead, making simulations with high rejection rates inefficient.

Chapter 4

Simulation model

In this chapter we discuss the chemical model that lies at the center of our simulation. We will first give the assumptions this model makes. Then we will discuss one of the properties included in our model, namely steric hindrance. Subsequently we will explain the probabilities of events that are included in our chemical model, which logically follow from these assumptions. Lastly we will use our chemical model to construct a kinetic scheme containing all possible reactions and their rate constants, which can be used in a KMC.

A chemical model is an abstract mathematical model of the chemical processes that occur during a simulation. Because it is not feasible to simulate every aspect of a chemical process in large-scale simulations, abstractions have to be made. In the case of polymers it is usually not relevant to simulate every single electron in the polymer, often not even every atom. In our research we aim to make the right abstractions so we include the most important aspects of the polymerization process.

4.1 Assumptions

Multiple assumptions about the chemical process have been made.

A 1 The chemical mixture is homogeneous, by both diffusion and manual stirring.

A 2 The only reactions which occur are radical-vinyl reactions. Termination reactions, i.e. radical-radical reactions, do not occur.

A 3 All reactive centers are, on average, active the same amount of time.

A 4 The reactivity of reactive groups is independent of the molecule in which they are located.

A 5 The volume, pressure and temperature of the reaction mixture do not change.

A 6 All reactive groups are available at the initial time. In other words, no reactive groups are created at a later time.

Assumptions A 1 through A 3 are based on the properties of RAFT polymerization. As mentioned before, RAFT polymerization creates a homogeneous mixture due to the slowdown of the polymerization process. Furthermore, in the presence of RAFT agents the amount of termination is negligible and the radicals are, on average, active for the same length of time. Assumption A 1 is especially important as this also implies that there are no localized effects. This assumption is central to simulations that do not track molecule locations, like ODE and KMC.

Assumption A 4 is based on the fact that the monomers and crosslinkers in copolymerization reactions are usually chosen in such a way that the local structure around the vinyl group and reactive centers are similar. As an effect, a reactive group on a monomer and crosslinker exhibits similar reactivity. We can conclude from this assumption in combination with Assumption A 3, that a molecule with twice the number of reactive groups is twice as reactive. Note that we do have to consider steric hindrance, so while reactive groups are equally reactive, they may not be equally accessible. In the following section we will describe the steric hindrance model we will use as part of our chemical model.

Assumption A 5 is based on the radical polymerization process, which usually happens in a controlled environment. The temperature is usually decided beforehand and the reaction mixture is kept at this temperature as closely as possible, making the effects of endothermic and exothermic reactions on the temperature of the reaction mixture negligible. Furthermore, liquid volume is a function of temperature and pressure, the latter of which is either set to regular air pressure or a vacuum. As such the volume of the reaction mixture will not change due to fluctuations in temperature and pressure. Note that we also assume that the volume of the reaction mixture is not changed by the reactions themselves. We base this on the fact that the largest part of the mixture is the solvent in which the initiators, monomers, crosslinkers and RAFT agent are dissolved. The change in volume due to the creation of polymers can thus be considered negligible in relation to the volume of the reaction mixture.

Assumption A 6 is based on the choice to only focus on fast and slow initiators, as mentioned in Section 1.2.3. Either the initiators are split instantaneously at the start of the polymerization and fully converted to initiator radicals, or the initiator is slow and contributes a negligible number of initiator radicals. Either way, we can assume that the number of initiators remains relatively unchanged over the course of the simulation. As termination also does not happen in our simulation, we can thus also conclude that the number of reactive centers remains the same. Note that we propose an extension of our simulation in Appendix C which allows the simulation of initiators that are neither considered fast nor slow.

4.2 Steric hindrance

As mentioned before in the previous section, even though reactive groups are considered equally reactive, they may not be equally accessible due to steric hindrance. Parts of the polymer itself can block access to reactive groups, as previously described in Section 1.2.5. Steric hindrance mainly has an effect on reactions between polymers, i.e. crosslinking reactions. The radical groups at the center of polymers are especially inaccessible to other polymers, as nearby polymer chains on both the radical and vinyl polymer can restrict access. Smaller molecules like the initiator radicals, monomers and crosslinkers do not undergo as much steric hindrance since the reactive groups themselves make up a large part of the molecule size.

We incorporate the steric hindrance model of Tripathi *& al.* as presented in [28] in our chemical model, to include the effects of steric hindrance. This model distinguishes different types of radical positions within a polymer. A radical at the end of a polymer chain has a different accessibility than one in the middle of a chain, resulting in a different rate coefficient. The specific positions are the following ones.

- **EN**: Chain end non-crosslinker radical. These are created when a monomer reacts with a radical, resulting in a monomer with a radical electron at the end of the polymer chain.
- **EC**: Chain end crosslinker radical. These are created when a crosslinker reacts with a radical,

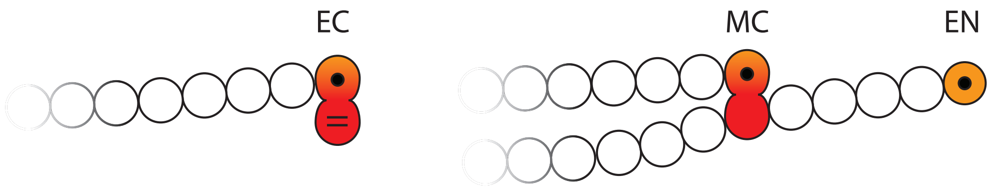


Figure 4.1: Schematic representation of possible radical positions. EC indicates a radical group on a chain end crosslinker, MC indicates a radical group on a mid-chain crosslinker and EN indicates a radical group on a chain-end non-crosslinker molecule.

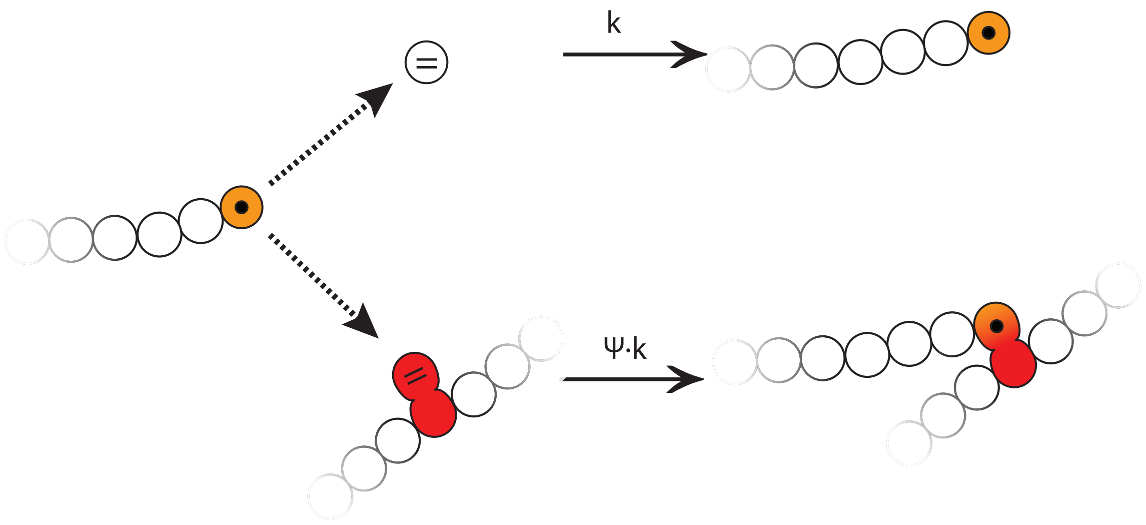


Figure 4.2: Schematic representation of propagation with a vinyl group in a crosslinker and a pendent vinyl group. The reaction rate of propagation with a pendent vinyl group has a factor ψ that is caused by the decreased accessibility of the vinyl group due to the polymer chain.

resulting in a crosslinker with a radical electron at the end of the polymer chain. The other side of the crosslinker will still have a vinyl group, which we refer to as a *pendent* vinyl group.

- **MC**: Mid-chain crosslinker radical. These are created when a pendent vinyl group reacts with a radical, resulting in a new crosslink.

The radical positions are shown schematically in Figure 4.1. The other reactive groups, vinyl groups, also undergo steric hindrance. A vinyl group in a polymer chain has a different accessibility than one in a monomer or crosslinker. This is caused by the monomers sticking out from the polymer backbone.

Factors on rate constants As both mid-chain radicals and pendent vinyl groups are considered to undergo steric hindrance, there are four possible reactions types to be considered:

1. a chain end radical reacts with vinyl group on a monomer or crosslinker
2. a chain end radical reacts with a pendent vinyl group
3. a mid-chain radical reacts with vinyl group on a monomer or crosslinker
4. a mid-chain radical reacts with a pendent vinyl group

Tripathi *et al.* were able to successfully model steric hindrance by expressing the rate constants of the latter three cases using a factor on the rate constant of the former. The rate constant of the first reaction type, k^{base}, thus functions as the base constant. The rate constant of the second reaction type is estimated using a factor ψ on this base rate: ψk^{base}. A schematic representation of this reaction type is shown in Figure 4.2. This factor is different for each monomer-crosslinker pair. The aim of the research was to find the values for ψ by modeling experimental data using a one-parameter fit. Tripathi *et al.* suggest that the success rate may depend on the length difference in monomer and crosslinker, as there seems to be a correlation; crosslinkers adjacent to longer monomers seem to undergo more steric hindrance [28].

The rate constant of the third reaction type is estimated using a factor ϕ on the base rate: ϕk^{base}. Tripathi *et al.* use the value for mid-chain radical propagation from the case of n-butyl acrylate FRP, which is 1.455×10^{-3} at 70° C [56].

The fourth reaction type is not included in the model of Tripathi *et al.* presumably due to the already low rate of mid-chain radical to non-pendent vinyl group reactions, which will be higher than the rate of mid-chain radical to pendent vinyl group reactions.

Like Tripathi, we will model steric hindrance for the four reaction types using a factor on the base rate constant.

4.3 Chemical model

The aforementioned assumptions combined with the steric hindrance model of Tripathi *et al.* lead to the following chemical model, in which only radical-vinyl reactions occur.

4.3.1 Notation

We will use the following notations in the description of our model:

- There are a total of K molecule species in the simulated reaction vessel.
- A molecule species $S_i \in \{I, M, C, P_4, P_5, ..., P_K\}$, with identifier $i \in \{1, 2, ..., K\}$ is either an initiator radical (I), a monomer (M), a crosslinker (C) or a polymer ($P_i, i \in \{4, 5, ..., K\}$).
- A reactive center has a position $Z \in \{EN, EC, MC\}$, corresponding to the radical positions described in the previous section.
- A molecule species S_i has several properties:
  - it contains a total number of $r_{Z,i}$ reactive centers in position Z
  - it contains a total number of
  $$r_i = \sum_{Z \in \{EN, EC, MC\}} r_{Z,i}$$
  reactive centers Note that the reactive centers in initiator radicals are in position EN.
  - it contains a total number of v_i vinyl groups
  - there are a total number of c_i molecules of this species in the simulated reaction vessel
- A molecule of polymer species $P_i = \langle I_i, M_i, C_i, V_i, EC_i, EC_i, MC_i \rangle$ contains:
  - I_i half-initiators
  - M_i monomers

  - C_i crosslinkers
  - V_i pendent vinyl groups
  - EN_i reactive centers in position EN
  - EC_i reactive centers in position EC
  - MC_i reactive centers in position MC

  Note that identifiers 1,2 and 3 are reserved for molecules species I, M and C, which are not polymers. Also note that $V_i = v_i$ and $Z_i = r_{Z,i}$ for each $Z \in \{EN, EC, MC\}$. A different notation is used for readability, making it clearer when we refer to elements of the tuple representing a molecule species.

- The reaction vessel has a total volume of vol_T.
- There are a total of $r_T = \sum_{i=1}^{K} c_i r_i$ reactive centers in the simulated reaction vessel.
- There are a total of $v_T = \sum_{j=1}^{K} c_j v_j$ vinyl groups in the simulated reaction vessel.
- k^{exp} is the experimentally obtained rate constant for a linear propagation reaction using a monomer of species M. k^{exp} is given in $\mathrm{l\,mol^{-1}\,h^{-1}}$, as is expected of a bimolecular reaction.

Furthermore, we will always notate the reaction equations in such a way that the molecule that functions as the radical is on the left-hand side of the plus and the molecule that functions as the vinyl is on the right-hand side.

Steric hindrance factor We use the aforementioned steric hindrance model of Tripathi *et al.* and extend it with the inclusion of mid-chain radical to pendent vinyl group reactions, which we assign a steric hindrance factor of ω. The reason for this extension is to create a more dynamic simulation which allows for the inclusion of these reactions.

We will use $f_{Z,j} \in [0, 1]$ to denote the steric hindrance factor on the base rate constant for a reaction between a radical in position Z and a vinyl group in a molecule of species S_j. The values for these factors are shown in Table 4.1 and the specific values for ψ, ϕ and ω will be discussed in Chapter 7.

Table 4.1: Steric hindrance factors $f_{Z,j}$

	$S_j \in \{M, C\}$	$S_j \in \{P_4, P_5, ..., P_K\}$
$Z \neq MC$	1	ψ
$Z = MC$	ϕ	ω

4.3.2 Rate constants of intermolecular reactions

From Assumption A 4 we concluded that a molecule with twice the number of reactive groups must be twice as reactive, when disregarding steric hindrance. This is the case for both radical groups and vinyl groups. We use the experimentally obtained rate constant, k^{exp}, of a propagation reaction in a linear polymerization, as our base rate constant. Note that in this propagation reaction both the radical and vinyl molecule only contain a single reactive group. When disregarding steric hindrance, we can express the rate constant of a reaction between a radical molecule of species S_i and a vinyl molecule of species S_j using this base rate constant as $r_i v_j k^{exp}$, essentially multiplying the base rate constant by the total number of possible combination between radical and vinyl groups. When we include the factor for steric hindrance $f_{Z,j}$ using the radical position Z, we obtain the rate constant

$$k_{Z,S_i,S_j} = f_{Z,j} r_{Z,i} v_j k^{exp} \tag{4.1}$$

for intermolecular reactions between radicals in position Z in the radical molecule species S_i and the vinyl molecule species S_j.

4.3.3 Rate constants of intramolecular reactions

The equation for the rate constant of intramolecular reactions is less straightforward. In this case we derive the rate constant by following the reaction steps of a crosslinked RAFT copolymerization.

- A reactive center in a molecule of species S_i becomes active by the detaching of a RAFT agent, forming a radical.
- The radical reacts with everything in its immediate environment.
- The reactive center becomes inactive again by attachment of a RAFT agent. Every reactive center is, on average, active for the same amount of time in the presence of a RAFT agent (Assumption A 3, as described in Section 4.1).

This 'immediate environment' can differ between molecules, even though the mixture is considered homogeneous. Our model includes the following differences:

- The same molecule cannot, by definition, be both the radical molecule and vinyl molecule in an intermolecular reaction, a case that becomes non-negligible once larger polymers are formed.
- The number of vinyl groups can differ between radical molecules, leading to a different number of possible intramolecular reactions.

Due to the differences in the immediate environments, different radicals can react with a different number of molecules in the time that they are active, even though they are active the same amount of time.

We created a model for the immediate environment using what we refer to as the *interaction radius* of the radical molecule, the radius in which it is able to react with other molecules. Within this radius can be both pendent vinyl groups of the radical molecule and vinyl groups on other molecules, as illustrated in Figure 4.3. Depending on the number of pendent vinyl groups in S_i and the concentration of vinyl groups on other molecules in the interaction radius, S_i either undergoes an intramolecular or intermolecular reaction. At low concentrations more intramolecular reactions occur [57].

Like the rate of intermolecular reactions, the rate of intramolecular reactions depends on concentrations, but in contrast to intermolecular reactions the rate of intramolecular reactions depends on the concentration of reactive groups within the interaction radius rather than the reaction vessel. To this purpose we introduce what we refer to as 'local concentration', which is the concentration of vinyl groups within the interaction volume of the radical molecule, defined as

$$[L_i] = \frac{v_i}{vol_i} \tag{4.2}$$

where vol_i is the (spherical) interaction volume with the interaction radius as the radius, given in liters, like vol_T. The function for vol_i is one of our parameters and will be further discussed in Chapter 7.

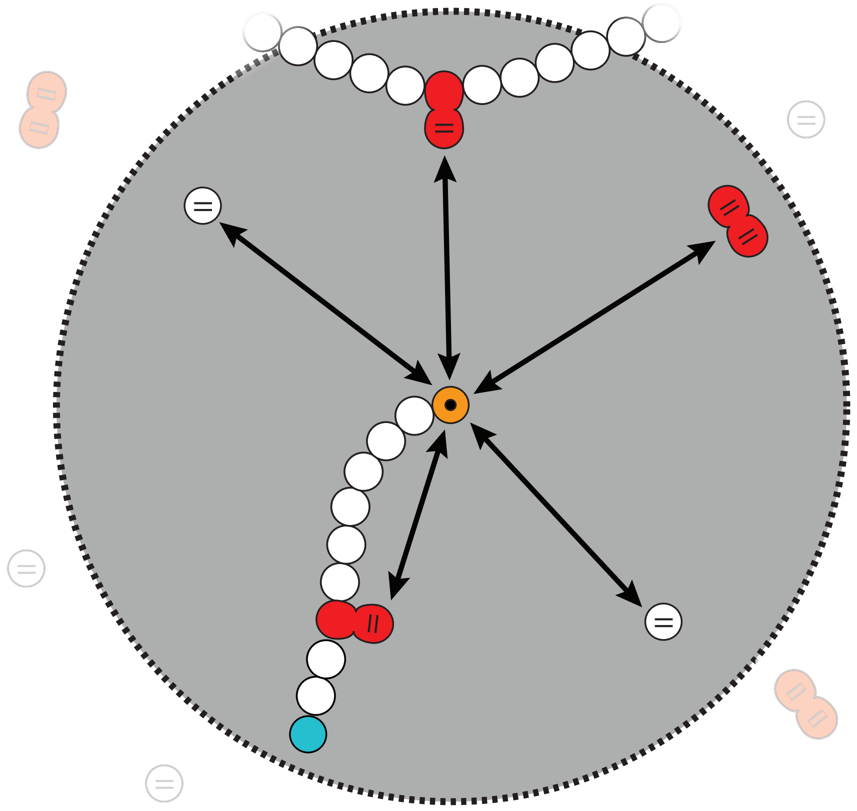


Figure 4.3: Schematic representation of interaction radius around radical group. The concentrations of vinyl groups within this radius decide the probability of intramolecular and intermolecular reactions. A lower concentration of vinyl groups leads to a higher average distance, thus lower probability of interaction between the reactive groups.

We also introduce 'global concentration', which is the concentration of vinyl groups not in the radical molecule, defined as

$$[G_i] = \frac{v_T - v_i}{vol_T}. \tag{4.3}$$

If all vinyl groups would be equally accessible, i.e. when we disregard steric hindrance, we would expect that the ratio of intermolecular to intramolecular reaction would be the same as the ratio of global concentration to local concentration. The probability of an intramolecular reaction with a radical molecule of species S_i in that case is

$$\frac{[L_i]}{[L_i] + [G_i]}. \tag{4.4}$$

Similarly, the probability of an intermolecular reaction in that specific case is

$$\frac{[G_i]}{[L_i] + [G_i]}. \tag{4.5}$$

The rate constant should reflect this and hence is chosen accordingly. When we include the factor for steric hindrance, which in the case of an intramolecular reaction can only be ψ or ω by definition, we get

$$k_{Z,S_i} = \frac{f_{Z,i} r_{Z,i} v_i k^{exp}}{vol_i A} \tag{4.6}$$

as the rate constant. This is effectively the same as the rate constant for an intermolecular reaction multiplied by a concentration of one vinyl molecule of species S_i per vol_i liter. In the following section we show that this rate constant gives us the expected ratio $[L_i]/[G_i]$, using the reaction rates obtained by from the rate constants.

Other simulations either use the same kinetic constant for all intramolecular reactions, use a time dependent coefficient for the probability of intramolecular crosslinking, or use the model for polymerization kinetics by Davis *et al.*, which also utilizes the interaction radius, but is far more complex than our model [2,28,48,58].

4.3.4 Reaction rates

As previously shown in Section 3.2, the reaction rates are used to calculate the reaction probabilities and are obtained by multiplying the number of molecules of the reactant species with the rate constants and, in the case of a bimolecular reaction, dividing by $vol_T A$. For intermolecular reactions between a radical molecule of species S_i with a radical in position Z and a vinyl molecule of species S_j the reaction rate is

$$R_{Z,i,j} = \frac{c_i c_j}{vol_T A} k_{Z,S_i,S_j} = \frac{c_i c_j}{vol_T A} f_{Z,j} r_{Z,i} v_j k^{exp} \tag{4.7}$$

where $i \neq j$ and

$$R_{Z,i,i} = \frac{c_i(c_i - 1)}{vol_T A} k_{Z,S_i,S_i} = \frac{c_i(c_i - 1)}{vol_T A} f_{Z,i} r_{Z,i} v_i k^{exp} \tag{4.8}$$

when both the radical and vinyl molecule are of species S_i. Both reaction rates are given in molecules h^{-1}. For the rate of intramolecular reactions of species S_i the reaction rate is

$$R_{Z,i} = c_i k_{Z,S_i} = c_i \frac{f_{Z,i} r_{Z,i} v_i k^{exp}}{vol_i A} \tag{4.9}$$

This gives us a total reaction rate of

$$R_T = \sum_{Z \in \{EC,EN,MC\}} \sum_{i=1}^{K} c_i r_{Z,i} \left(\frac{v_i f_{Z,i}}{vol_i} + \frac{\sum_{j \neq i} c_j v_j f_{Z,j}}{vol_T} + \frac{(c_i - 1) v_i f_{Z,i}}{vol_T} \right) \frac{k^{exp}}{A} = \sum_{Z \in \{EC,EN,MC\}} \sum_{i=1}^{K} c_i r_{Z,i} \left(\frac{v_i f_{Z,i}}{vol_i} + \frac{\sum_{j=1}^{K} c_j v_j f_{Z,j}}{vol_T} - \frac{v_i f_{Z,i}}{vol_T} \right) \frac{k^{exp}}{A} \tag{4.10}$$

In Section 4.3.3 we mentioned that, under the assumption that there is no steric hindrance, the ratio between intramolecular and intermolecular reactions should be the same as the ratio between local and global concentrations. Let us again consider a radical molecule of species S_i. When disregarding steric hindrance we would thus expect

$$\sum_{Z \in \{EN,EC,MC\}} \frac{\overline{R_{Z,i}}}{\sum_{j=1}^{K} \overline{R_{Z,i,j}}} = [L_i][G_i]^{-1},$$

where $\overline{R_i}$ and $\overline{R_{i,j}}$ are $R_{Z,i}$ and $R_{Z,i,j}$ with $f_{Z,i}$ substitued by 1 respectively, which we show is indeed the case in Equation D.1 in Appendix D.

4.4 Kinetic scheme

The reactions in our kinetic scheme are limited to radical-vinyl reactions that cause permanent changes to polymer structures. We exclude reactions with RAFT agents, but do include the effects caused by the presence of RAFT agent, such as negligible termination reactions, a homogeneous mixture and similar activity of radical groups. The slowdown caused by RAFT reactions is not included, as our implementation does not include a concept of time, which will be further discussed in Section 5.1. We will use the previously described notation $P_i = \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle$ to describe a polymer species and I, M and C for the initiator radicals, monomers and crosslinkers species respectively. The rate constants k_{Z,S_i,S_j} and k_{Z,S_i} are as defined in Sections 4.3.2 and 4.3.3. Our kinetic scheme contains the following reactions.

Initiation

$$I + M \xrightarrow{k_{EN,I,M}} \langle 1,1,0,0,1,0,0 \rangle \tag{4.11}$$

$$I + C \xrightarrow{k_{EN,I,C}} \langle 1,0,1,0,0,1,0 \rangle \tag{4.12}$$

$$I + P_i \xrightarrow{k_{EN,I,P_i}} \langle I_i + 1, M_i, C_i, V_i - 1, EN_i, EC_i, MC_i + 1 \rangle \tag{4.13}$$

Propagation with monomer

$$P_i + M \xrightarrow{k_{EN,P_i,M}} \langle I_i, M_i + 1, C_i, V_i, EN_i, EC_i, MC_i \rangle \tag{4.14}$$

$$P_i + M \xrightarrow{k_{EC,P_i,M}} \langle I_i, M_i + 1, C_i, V_i, EN_i + 1, EC_i - 1, MC_i \rangle \tag{4.15}$$

$$P_i + M \xrightarrow{k_{MC,P_i,M}} \langle I_i, M_i + 1, C_i, V_i, EN_i + 1, EC_i, MC_i - 1 \rangle \tag{4.16}$$

Propagation with crosslinker

$$P_i + C \xrightarrow{k_{EN,P_i,C}} \langle I_i, M_i, C_i + 1, V_i + 1, EN_i - 1, EC_i + 1, MC_i \rangle \tag{4.17}$$

$$P_i + C \xrightarrow{k_{EC,P_i,C}} \langle I_i, M_i, C_i + 1, V_i + 1, EN_i, EC_i, MC_i \rangle \tag{4.18}$$

$$P_i + C \xrightarrow{k_{MC,P_i,C}} \langle I_i, M_i, C_i + 1, V_i + 1, EN_i, EC_i + 1, MC_i - 1 \rangle \tag{4.19}$$

Intermolecular crosslinking

$$P_i + P_j \xrightarrow{k_{EN,P_i,P_j}} \langle I_i + I_j, M_i + M_j, C_i + C_j, V_i + V_j - 1, EN_i + EN_j - 1, EC_i + EC_j, MC_i + MC_j + 1 \rangle \tag{4.20}$$

$$P_i + P_j \xrightarrow{k_{EC,P_i,P_j}} \langle I_i + I_j, M_i + M_j, C_i + C_j, V_i + V_j - 1, EN_i + EN_j, EC_i + EC_j - 1, MC_i + MC_j + 1 \rangle \tag{4.21}$$

$$P_i + P_j \xrightarrow{k_{MC,P_i,P_j}} \langle I_i + I_j, M_i + M_j, C_i + C_j, V_i + V_j - 1, EN_i + EN_j, EC_i + EC_j, MC_i + MC_j \rangle \tag{4.22}$$

Intramolecular crosslinking

$$P_i \xrightarrow{k_{EN,P_i}} \langle I_i, M_i, C_i, V_i - 1, EN_i - 1, EC_i, MC_i + 1 \rangle \tag{4.23}$$

$$P_i \xrightarrow{k_{EC,P_i}} \langle I_i, M_i, C_i, V_i - 1, EN_i, EC_i - 1, MC_i + 1 \rangle \tag{4.24}$$

$$P_i \xrightarrow{k_{MC,P_i}} \langle I_i, M_i, C_i, V_i - 1, EN_i, EC_i, MC_i \rangle \tag{4.25}$$

Chapter 5

Simulation methods

In this chapter we discuss our Rejection Kinetic Monte Carlo (RKMC) algorithm. We use the kinetic scheme presented in Section 3.1 as the basis of our KMC simulation, which describes the possible reactions and the associated reaction rates.

Firstly, we discuss the major differences from a traditional KMC and will explain why our approach fits the simulation of polymerization reactions better. Secondly we discuss the algorithm itself and how it follows from our chemical model. Section 5.2 follows the steps of the RKMC algorithm as described in Chapter 3, namely initialization, reaction selection, reaction acceptation, time step calculation, reaction simulation and iteration. Thirdly, in Section 5.3, we will explain the Binary Indexed Tree (BIT), a data structure that allows us to do sampling efficiently in our simulation. Lastly, in Section 5.4 we discuss the graph models that are used to describe polymer structures.

5.1 Main differences from KMC

Our simulation algorithm differs on multiple points from a traditional KMC algorithm. The three major differences are the absence of time, the tracking of particle reactivity rather than the reaction rate of each possible reaction and the combining of sampling techniques from both the KMC and RKMC algorithms. All other differences are a result of these three differences.

Absence of time Requirement R 5, given in Section 1.4, states that our simulation should be able to predict properties *at a certain point of conversion* rather than at a point in time. It was not necessary to also track time in our simulation. This allowed us to gain performance by leaving out other properties of the simulation as well.

The exclusion of RAFT reactions has been discussed in Section 4.4 where we explain the kinetic scheme. Structural changes caused by the reversible reactions between active centers and a RAFT agents are not permanent, and, as such not included in our simulation. For the inclusion of these RAFT reactions we refer to the extension of our algorithm, as presented in Appendix C. Most effects, including a homogeneous chemical mixture, absence of termination reactions and radical activity have been used in the design of the kinetic scheme.

The one effect of the RAFT agent that has not been included via the kinetic scheme yet, is the addition of extra initiator radicals via the RAFT agent, as based on assumption A 6. These additional initiator radicals have been taken into account in our KMC algorithm and are further discussed in Section 5.2.1.

S_j \ S_i	S_1	S_2	$\cdots$	S_K
$\varnothing$	R_1	R_2	$\cdots$	R_K
S_1	$R_{1,1}$	$R_{2,1}$	$\cdots$	$R_{K,1}$
S_2	$R_{1,2}$	$R_{2,2}$	$\cdots$	$R_{K,2}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
S_K	$R_{1,K}$	$R_{2,K}$	$\cdots$	$R_{K,K}$

Figure 5.1: Sampling table for selection of reaction $S_i + S_j \rightarrow S_k$ with radical species S_i, vinyl species S_j and rate $R_{i,j}$. Each reaction has a probability of $R_{i,j}R_T^{-1}$ to be selected where R_T is again the total sum of reaction rates. Reactions where $S_j = \varnothing$ are intramolecular reactions.

Reactivity instead of rates A traditional KMC simulation keeps track of all reaction rates. This is a rather inefficient approach when simulating highly crosslinked RAFT polymerization, as most molecule species contain both pendent vinyl groups and reactive centers. As a consequence most species can react with each other, making the simulation with K molecule species approach the worst-case number of possible reactions: $\mathcal{O}(K^2)$. The sampling table with all possible combinations of molecule species has been visualized in Figure 5.1. It is clear that every time the number of molecules of a species changes, the rate of each possible reaction in which that species can participate, either as a vinyl or a radical molecule, has to be updated. This means that one row and one column of the sampling table have to be updated.

By tracking the reactivities of molecule species, with which we can calculate the reaction rates, we only need to keep track of $\mathcal{O}(K)$ values, greatly improving the scaling of the KMC algorithm. Not only can we sample from a sampling table of size $\mathcal{O}(K)$, as we show in Section 5.2.2, but updating the sampling table can also be done significantly faster, as only the reactivity of each molecule species has to be updated rather than the rates of all reactions in which the species is either a product or a reactant. We go into more detail about the latter in Section 5.2.4.

Combination of KMC and RKMC techniques Our algorithm uses a rather naive implementation for steric hindrance. Rather than including the factor steric hindrance introduces to the reaction rate, we omit this factor in the reaction selection step of the KMC algorithm. This effectively yields the reaction rates given in Equations 4.7 through 4.9 with $f_{Z,i} = 1$. Other than this change in reaction rates, the selection of the reaction still follows the rejection-free KMC algorithm. To correct for the omission of the steric hindrance factor we include the reaction acceptation step of the RKMC algorithm, where $f_{Z,i}$ functions as the success rate of our reaction. Our algorithm thus combines the reaction selection of a rejection-free KMC and the reaction acceptation of a RKMC.

5.2 Algorithm

In this section we explain the steps our KMC algorithm performs. We will use the order of steps of traditional RKMC.

5.2.1 Initialization

The initial state of the simulation is created during the initialization step of the algorithm. This is done with the help of the given simulation parameters, which are entered by the user. The simulation algorithm itself needs the usual parameters of a KMC simulation, which are the concentrations $[S_i]$ of all initial molecule species S_i in the simulation, as well as the total number of molecules c_T in the simulation. The additional parameters that are needed are the steric hindrance factors ψ, ϕ and ω, and the function for the interaction volume of a polymer vol_i.

We follow the algorithm presented in Chapter 3 to obtain the number of molecules c_i of each molecule species S_i using the related concentration $[S_i]$. A typical simulation of crosslinked RAFT copolymerization starts with a number of initiator radicals, I, a number of crosslinkers, C, and a number of monomers, M. Note that $[I]$ contains the initiator radicals introduced via the RAFT agent, and, in the case of fast-splitting initiators, the initiator radicals introduced via the initiator as well. The total volume of the simulated vessel vol_T is also obtained as described in Chapter 3.

Using the calculated molecule numbers we can populate the simulation. We create the initial state, which consists of:

- **Molecule species**: An ordered set of species $\{S_1, S_2, ..., S_K\}$, each described by a tuple $S_i = \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle$, containing the same elements as our polymer tuple $P_i = \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle$ in the kinetic scheme. We use $I = \langle 1, 0, 0, 0, 1, 0, 0 \rangle$ for initiator radicals, $M = \langle 0, 1, 0, 1, 0, 0, 0 \rangle$ for monomers and $C = \langle 0, 0, 1, 2, 0, 0, 0 \rangle$ for crosslinkers.

- **Quantities**: An ordered set of molecule counts $\{c_1, c_2, ..., c_K\}$, in one-to-one correspondence with the molecule species.

- **Binary Indexed Trees (BITs)**: BITs are data structures that can be used to calculate prefix sums of a list of non-negative numbers in logarithmic time. The data structure itself is an array that is interpreted as a binary tree, in which each element is a partial sum of the numbers in the list. We use the BITs to track the reactivity of molecule species. The exact properties that are tracked are further described in Section 5.2.2 and BITs will be discussed in more detail in Section 5.3.

- **Abstraction map**: A map $\{a_d(S_1) \mapsto \overline{\mathcal{S}_1}, a_d(S_2) \mapsto \overline{\mathcal{S}_2}, ..., a_d(S_K) \mapsto \overline{\mathcal{S}_K}\}$ from abstractions of molecule species to the structural model of the species. The purpose and implementation of the abstraction function $a_d(S_i)$ will be further discussed in Section 5.2.4 and the structural model $\overline{\mathcal{S}_i}$ will be further discussed in Section 5.4.

- **Number of vinyl groups**: The total number of vinyl groups, v_T, that have not yet reacted, calculated from the initial number of monomers and crosslinkers.

Once the initial state has been created using the user-given parameters, the simulation can begin. The next step in the simulation algorithm is reaction selection.

5.2.2 Reaction selection

A reaction is selected in every iteration of the KMC algorithm using the sampling table shown in Figure 5.1. As explained in Section 3.2, the probability of a reaction to be chosen for simulation is its rate divided by the sum of all reaction rates. There are two problems that need to be addressed.

- The sampling table of a KMC simulation scales quadratically with the number of molecule species, as shown in Section 5.1.

- No efficient algorithm for sampling from the sampling table and updating the sampling table is provided by the KMC algorithm.

In this section we will show how the tracking of the reactivity of molecule species can be used to reduce the sampling table such that it scales linearly with the number of molecule species. At the same time we will show how multiple BITs can be used to efficiently select reactions from the sampling table.

Multistage sampling Rather than sampling the sampling table as a whole, we use multistage sampling, which is a sampling process in which samples are taken from a smaller subset of the sampling space at each stage. We split up the sampling table by radical species, number of reactants, vinyl species and radical position, as illustrated in Figure 5.2. By sampling at each stage we select a radical species S_i, an intramolecular or intermolecular reaction, a vinyl species S_j and a radical position Z respectively.

Before showing how we perform multistage sampling, we first substitute $f_{Z,j}$ by 1, as explained in Section 5.1. We obtain the reaction rates

$$\overline{R_{i,j}} = \frac{c_i c_j}{vol_T A} r_i v_j k^{exp} \tag{5.1}$$

where $i \neq j$ and

$$\overline{R_{i,i}} = \frac{c_i(c_i - 1)}{vol_T A} r_i v_i k^{exp} \tag{5.2}$$

for intermolecular reactions and

$$\overline{R_i} = c_i \frac{r_i v_i k^{exp}}{vol_i A} \tag{5.3}$$

for intramolecular reactions, giving a total sum of reaction rates

$$\overline{R_T} = \sum_{i=1}^{K} \overline{R_i} + \sum_{j=1}^{K} \overline{R_{i,j}} = \sum_{i=1}^{K} c_i r_i \left(\frac{v_i}{vol_i} + \frac{\sum_{j=1}^{K} c_j v_j}{vol_T} - \frac{v_i}{vol_T} \right) \frac{k^{exp}}{A}, \tag{5.4}$$

which can also be written as

$$\overline{R_T} = \left(\sum_{i=1}^{K} \frac{c_i r_i v_i}{vol_i} + vol_T^{-1} \left(v_T \sum_{i=1}^{K} c_i r_i - \sum_{i=1}^{K} c_i r_i v_i \right) \right) \frac{k^{exp}}{A} \tag{5.5}$$

where v_T is the total number of vinyl groups in the simulation, like defined in Section 4.3. Note that this number is part of our simulation state and does not need to be calculated. The first sum in Equation 5.5 represents the intramolecular reactions and the second the intermolecular reactions. The third sum is a correction of the second sum, essentially excluding the case where the same molecule is both the radical and vinyl in an intermolecular reaction, which is impossible by definition. We will now explain how this equation can be used for reaction selection.

Radical selection The first step in the selection of a reaction is selecting the radical molecule species that is going to react. This can be done by using Equation 5.5. We keep track of the prefix

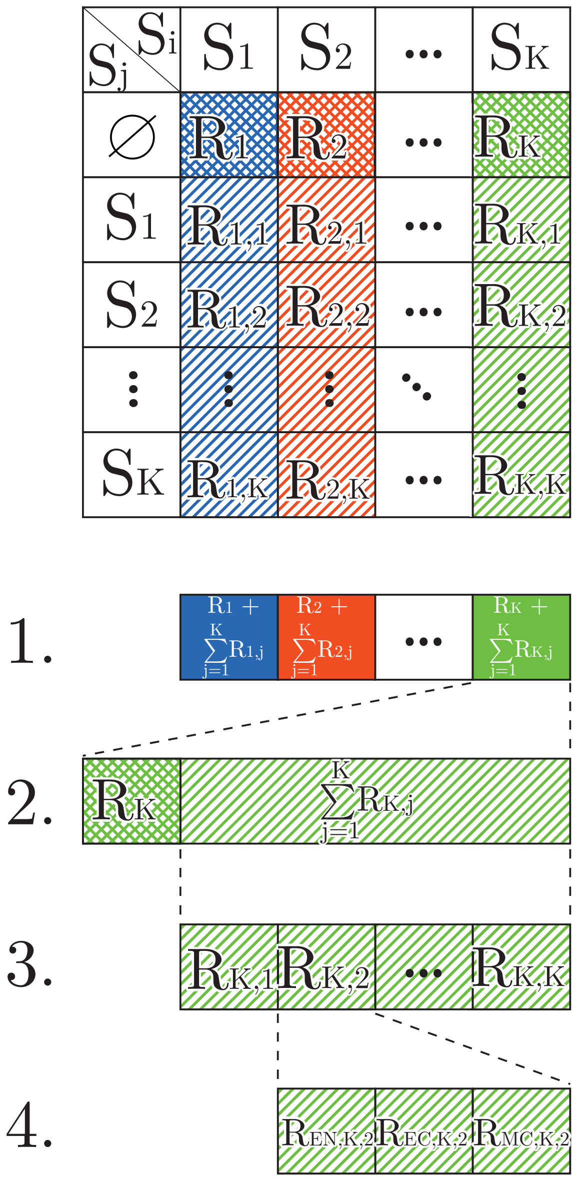


Figure 5.2: We perform multistage sampling by splitting up the sampling space, which is shown in Figure 5.1, by 1.) the radical species S_i, 2.) reaction type (intramolecular and intermolecular) 3.) vinyl species S_j (for intermolecular reactions) and 4.) the radical position $Z \in \{EN, EC, MC\}$. At each stage a sample is taken with the probability of each option depending on the reaction rates, yielding a reaction identified by Z, i and j.

sums

$$\sigma_j^{intra} = \sum_{i=1}^{j} \frac{c_i r_i v_i}{vol_i}, \quad \sigma_j^{inter} = \sum_{i=1}^{j} c_i r_i, \text{ and } \sigma_j^{corr} = \sum_{i=1}^{j} c_i r_i v_i$$

with $1 \leq j \leq K$, using the previously mentioned BITs. $intra$, $inter$ and $corr$ stand for intramolecular, intermolecular and correction respectively, indicating the purpose the BITs serve in calculating the sum of reactivities. Similar to Equation 3.9 we select a random reaction by generating a uniformly distributed random number x in the interval $[0, 1)$, and subsequently finding i such that

$$\sigma_{i-1}^{intra} + \frac{v_T \sigma_{i-1}^{inter} - \sigma_{i-1}^{corr}}{vol_T} < x \left(\sigma_K^{intra} + \frac{v_T \sigma_K^{inter} - \sigma_K^{corr}}{vol_T} \right) < \sigma_i^{intra} + \frac{v_T \sigma_i^{inter} - \sigma_i^{corr}}{vol_T}. \tag{5.6}$$

This i is the identifier of the selected radical species S_i. In Section 5.3 we refer to the algorithm presented by Fenwick in [59] for finding the index of a prefix sum by performing a binary search. This algorithm has been extended to perform the binary search over the three BITs in parallel.

Intra/intermolecular reaction selection Once we have chosen a random radical species S_i we can select the vinyl species. The first choice that has to be made is between an intramolecular and an intermolecular reaction. The process for this is relatively straightforward. We have argued previously in Section 4.3.3 that the probability of an intramolecular reaction is $[L_i]([L_i]+[G_i])^{-1}$ when disregarding the steric hindrance factor $f_{Z,j}$. We can thus simply generate a uniformly distributed random number x in the interval $[0, 1)$ and choose an intramolecular reaction if

$$x([L_i] + [G_i]) < [L_i]$$

and an intermolecular reaction otherwise. In the case of the former, the vinyl species is also S_i. In the case of the latter, the exact vinyl species still has to be chosen.

Vinyl selection For the selection of a vinyl molecule we introduce a fourth BIT, simply tracking the prefix sums

$$\sigma_j^{vinyl} = \sum_{i=1}^{j} c_i v_i$$

with $1 \leq j \leq K$. Note that $\sigma_K^{vinyl} = v_T$. In Equation D.2 in Appendix D we show that

$$\overline{R_{i,j}}\ \overline{R_T}^{-1} = \frac{c_j v_j}{v_T - v_i}$$

when $i \neq j$ and

$$\overline{R_{i,i}}\ \overline{R_T}^{-1} = \frac{(c_i - 1) v_j}{v_T - v_i}$$

otherwise. This ratio is equal to the probability that species S_j is selected. We again generate a uniformly distributed random number x in the interval $[0, 1)$ and calculate $y = x(v_T - v_i)$. If $y < \sigma_i^{vinyl} - v_i$ we calculate j for

$$\sigma_{j-1}^{vinyl} < y < \sigma_j^{vinyl} \tag{5.7}$$

and otherwise for

$$\sigma_{j-1}^{vinyl} < y + v_i < \sigma_j^{vinyl}, \tag{5.8}$$

essentially omitting v_i from our sampling space v_T. This is necessary to compensate for the fact that an intramolecular reaction has already been ruled out.

Radical position selection As a last step in the reaction selection process we randomly choose position Z by generating a uniformly distributed random number x in the interval $[0,1)$ and calculating $y = x(r_{EN,i} + r_{EC,i} + r_{MC,i})$. We then choose Z using

$$Z = \begin{cases} EN & \text{if } y < r_{EN,i} \\ EC & \text{if } r_{EN,i} \leq y < r_{EN,i} + r_{EC,i} \\ MC & \text{otherwise} \end{cases}$$

At this point we have selected either an intramolecular reaction for species S_i or an intermolecular reaction between S_i and S_j with a radical in position Z. In the next steps of the algorithm the values for i, j and Z are fixed until reaction selection is performed again.

In summary, we can thus sample a reaction by tracking vol_T and c_i, r_i, v_i and vol_i for each of the K molecule species. The BITs keep track of the prefix sums σ_j^{intra},σ_j^{inter},σ_j^{corr} and σ_j^{vinyl} with $1 \leq j \leq K$, which are the sums of the reactivities of the molecule species, as defined in Section 5.2.2.

Since only these sums need to be tracked to perform sampling, our sampling table scales linearly with K. When we combine this with the ability to obtain a prefix sum in $\mathcal{O}(log(n))$ with a BIT mapped to a list of n values, we can thus perform reaction sampling in $\mathcal{O}(log(K))$.

5.2.3 Reaction acceptance

As mentioned in the previous section, our sampling algorithm gives an overestimation of the total reaction rate in the simulation: it does not account for steric hindrance. To correct for the steric hindrance we perform a simple sampling procedure using the success rate $f_{Z,j}$, as defined in Section 4.3, with Z and j as chosen by our reaction selection. We generate a uniformly distributed random number x in the interval $[0,1)$ and reject the reaction if $x \geq f_{Z,j}$. If the reaction is rejected the algorithm simply returns to the reaction selection step.

It is clear that this step has a negative effect on performance, especially at lower values of $f_{Z,j}$, where many reactions would be selected, but only few would be simulated. This is confirmed by our test results, as we show in Section 7.4, which is why we suggest an extension to our algorithm which is rejection-free. This extension is presented in Appendix C.

5.2.4 Reaction simulation

Once a reaction is accepted it will be simulated. Reaction simulation follows the reaction scheme described in Section 3.1. The reaction rule $\mathcal{R}_h$ associated with the reaction between the previously selected species is selected from the kinetic scheme. If a single species is selected we will simulate an intramolecular reaction using $\mathcal{R}_h : S_i \to S_k$. Otherwise, when reaction selection yields two species we will use the reaction rule $\mathcal{R}_h : S_i + S_j \to S_k$ to simulate an intermolecular reaction.

The reaction rules in the kinetic scheme are easily translated to the tuples which represent molecule species. Note that these rules do not describe the alteration and combination of the molecular structures of these species. To this purpose we have defined an additional set of reaction rules, which will be further discussed in Section 5.4. The inclusion of molecular structures is optional and is disabled by default.

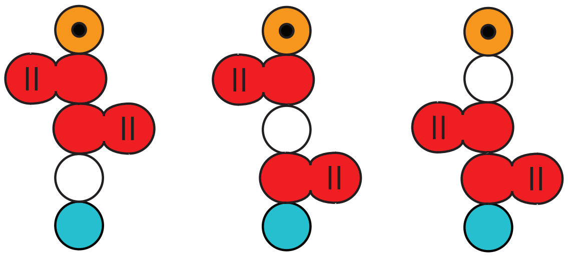


Figure 5.3: Schematic representation of multiple structures that are possible for $S_i = \langle 1, 2, 2, 2, 1, 0, 0 \rangle$.

We have to consider an additional problem: for each molecule species represented by our tuple there may be multiple structures, as illustrated by Figure 5.3. It is simply not feasible to keep track of each structure that can be encountered in our simulation. As a solution to this problem we limit the number of structures that are tracked. We allow for one structure per species or one structure shared by multiple species, the latter allows us to further reduce the number of structures, the creation of which is, as we show in Section 5.4, an expensive operation. As a trade-off to this reduction, the structures become more uniform due to a decrease in structural variety.

We will now show how the limitation of molecule structures is achieved. We define abstraction functions $a_d(S_x)$ with detail level d, mapping the species tuple S_x to an abstraction of that tuple. In essence, $a_d(S_x)$ is the equivalence class for all species $\{S_y | a_d(S_x) = a_d(S_y)\}$. We will use the first species S_x encountered in each equivalence class as the *representative* of this class.

By combining the abstraction functions for molecule species and reactions with the abstraction map specified in Section 5.2.1, it is possible to find out if the molecule structure of the equivalence class has been calculated before. Every time a new molecule species S_k is created, we simply check whether an existing entry $a_d(S_k) \rightarrow \overline{S_k}$ exists in the abstraction map. If an entry already exists in the abstraction map, no new structure is calculated. If there is no such entry, we apply the abstraction function to the tuples representing the reactant species and obtain their respective structures from the abstraction map. $a_d(S_i) \rightarrow \overline{S_i}$, and, in the case of an intermolecular reaction, $a_d(S_j) \rightarrow \overline{S_j}$ as well. These structures are then combined into $\overline{S_k}$, the structure of the product species, using the reaction rules given in Section 5.4. Finally, the entry $a_d(S_k) \rightarrow \overline{S_k}$ is added to the abstraction map.

Detail levels We have defined two detail levels:

1. **High detail**: On high detail level we consider all molecules to be equivalent that have the same number of half-initiators, monomers, crosslinkers, vinyl groups, and active centers in the EN, EC and MC positions. High detail thus includes molecular composition, reactivity and location of reactive groups. On this detail level the abstraction function simply returns the tuple unaltered, i.e.
   $a_{high} : \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle \mapsto \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle$.

2. **Low detail**: On low detail all molecules with equivalent number of subunits (monomers, crosslinkers and half-initiators), vinyl groups and active centers in the EN, EC and MC positions are considered equivalent. On this detail level molecule size (in terms of number of subunits), reactivity and location of reactive groups are included. The abstraction function for this equivalence class is
   $a_{low} : \langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle \mapsto \langle I_i{+}M_i{+}C_i, V_i, EN_i, EC_i, MC_i \rangle$.

More abstraction than a_{low} is not recommended, as the location of reactive groups is used by the sampling algorithm for simulations with molecule structures, as described in Section 5.4.3.

Update state Once the reaction product S_k of the simulated reaction is known the simulation state can be updated. The following changes are made to the simulation state.

- **Molecule species**: If S_k is a molecule species that has not been encountered before, we add it to the set of molecule species. This thus increases the total number of species K.
- **Quantities**: The number of the molecules of the reactant species is decreased by one and the number of molecules of the product species is increased by one, i.e. $c_i := c_i - 1$, $c_j := c_j - 1$ and $c_k := c_k + 1$.
- **BITs**: The reactivity of species S_i, S_j and S_k is updated accordingly by adjusting the BITs, using the function described in Section 5.3.
- **Number of vinyl groups**: The number of vinyl groups is decreased by one, i.e. $v_T := v_T - 1$.

5.2.5 Iteration

The iteration step of our algorithm is straightforward. The halting criteria are checked and if none yield true, the algorithm goes back to the reaction selection step. If one of the halting criteria holds, the necessary data is collected from the simulation state and the simulation itself halts. The three main halting criteria are:

- Reaching 100% conversion.
- Only having a single polymer left and no initiator radicals. This is associated with reaching the gel point; the further growth of the gel by reactions with residual monomers and crosslinkers is not interesting and as such is not simulated.
- Maximum polymer size for 3D model generation has been reached. 3D models cannot often not be used if they exceed a certain size, so simulation is halted once this size, as entered by the user, is reached. As structured simulations mainly serve to generate 3D models for visual feedback they are halted once no more 3D models need to be generated. These maximum sizes will be further discussed in Section 6.3.2.

5.3 Binary Indexed Trees

The KMC algorithm does not describe the data structures that should be used to track reactivity rates. Naive implementations have sufficient average-case time complexities for sampling, but bad worst-case time complexities. This makes the performance of these implementations dependent on simulation scenarios. A clear case of this is the use of simple lists for tracking the reaction rates; to find the prefix sum of these rates, which is necessary when sampling, as indicated by Equation 3.9, it is possible to simply iterate over the list and keep track of the prefix sum of these rates. Even though it is possible to update a reaction rate in $\mathcal{O}(1)$ time, calculating the prefix sum has a worst-case time complexity of $\mathcal{O}(n)$. The average-case time complexity can be improved by ordering the list in such a way that the reactions with the biggest reaction rates are at the start of this list. Though this approach is sufficiently efficient in simulations with few possible reactions, it becomes inefficient when simulating a large number of reactions with similar reaction rates, as is the case in (highly) crosslinked polymerization. It is thus worth investigating data structures that improve the time performance of calculating prefix sums.

The Fenwick tree, also known as the Binary Indexed Tree (BIT), is a data structure designed specifically to calculate prefix sums of lists containing non-negative numbers [59]. An example of a BIT is shown in Figure 5.4 and an example of how we can calculate prefix sums is shown in Figure 5.6. BITs are arrays interpreted as binary trees, making them memory-efficient with a memory complexity of $\mathcal{O}(n)$ for a tree of n nodes. A BIT allows us to both update values and calculate prefix sums in $\mathcal{O}(log(n))$. Both the algorithms for updating the BIT and calculating prefix sums are described by [60] with some explanatory examples. Furthermore, Fenwick has defined a binary search function that finds the element corresponding to a given prefix sum. This allows us to find the value of i in Equation 3.9 in $\mathcal{O}(log(n))$ time.

BITs of size n are saved in an array of size $n+1$. The first element of this array is always zero and its function is to make the BIT essentially one-indexed. This allows for a more efficient use of bit-wise operations, which we will not further discuss in this work.

5.3.1 Optimization

We created multiple algorithms to keep the BITs as small as possible for better efficiency. The BITs only track the reactivity of species S_i for which $c_i > 0$, as species of which there are no molecules cannot undergo reactions. Since some species are not tracked, BITs do not always map one-to-one to the set of molecule species $\{S_1, S_2, ..., S_k\}$. We will now explain how we *do* map the BITs to the molecule species and their reactivities using the following definitions and structures:

- We define the set of non-empty molecule species $\mathcal{H} = \{S_i | S_i \in \{S_1, S_2, ..., S_k\}, c_i > 0\}$.
- We define the number of non-empty molecule species $\overline{K} = |\mathcal{H}|$.
- The array $a[]$ is used to track the reactivities of the non-empty molecule species and the array $BIT[]$ is the BIT that is used to track the prefix sums of $a[]$. Like $BIT[]$, $a[]$ has an unused zero as the first element as to easily allow one-indexing.
- We keep track of variable $b = 2^{\lfloor \log_2 \overline{K} \rfloor + 1} + 1$, which is the number of non-empty molecule species rounded up to the first power of two, plus one. This b will be the size of the arrays. The plus one is because of the extra element at index zero. It is important to note that the size of the $BIT[]$ array is kept at one larger than a power of two, so the total sum is always saved in the last position of the array, as shown in Figure 5.4.
- We use the function $adj(m, x)$ as defined by [60], which updates the BIT after the value of $a[m]$ is increased by x.
- We use the function $getIndex(x, r)$ as defined by [59], which utilizes a binary search through the BIT, starting at the node with index r, to obtain i for $rsq(i-1) < x < rsq(i)$. We will use $r = b - 1$, as the root of the tree is always located at the last index of the BIT.
- We keep track of the injective map $index : \{j | S_j \in \mathcal{H}\} \to \{1, 2, ..., b-1\}$ which maps identities of all non-empty species to an index of the arrays $BIT[]$ and $a[]$.
- We define the partial mapping $id : \{1, 2, ..., b-1\} \to \{j | S_j \in \mathcal{H}\}$, which is the inverse map of $index$.
- We define the set of unused indices $\mathcal{E} = \{m | m \in \{1, 2, ..., b-1\}, m \notin dom(id)\}$ in the arrays.
- We define the ordered list of these indices $\overline{\mathcal{E}} = (m_1, m_2, ..., m_{|\mathcal{E}|})$ containing all elements in $\mathcal{E}$ in ascending order.

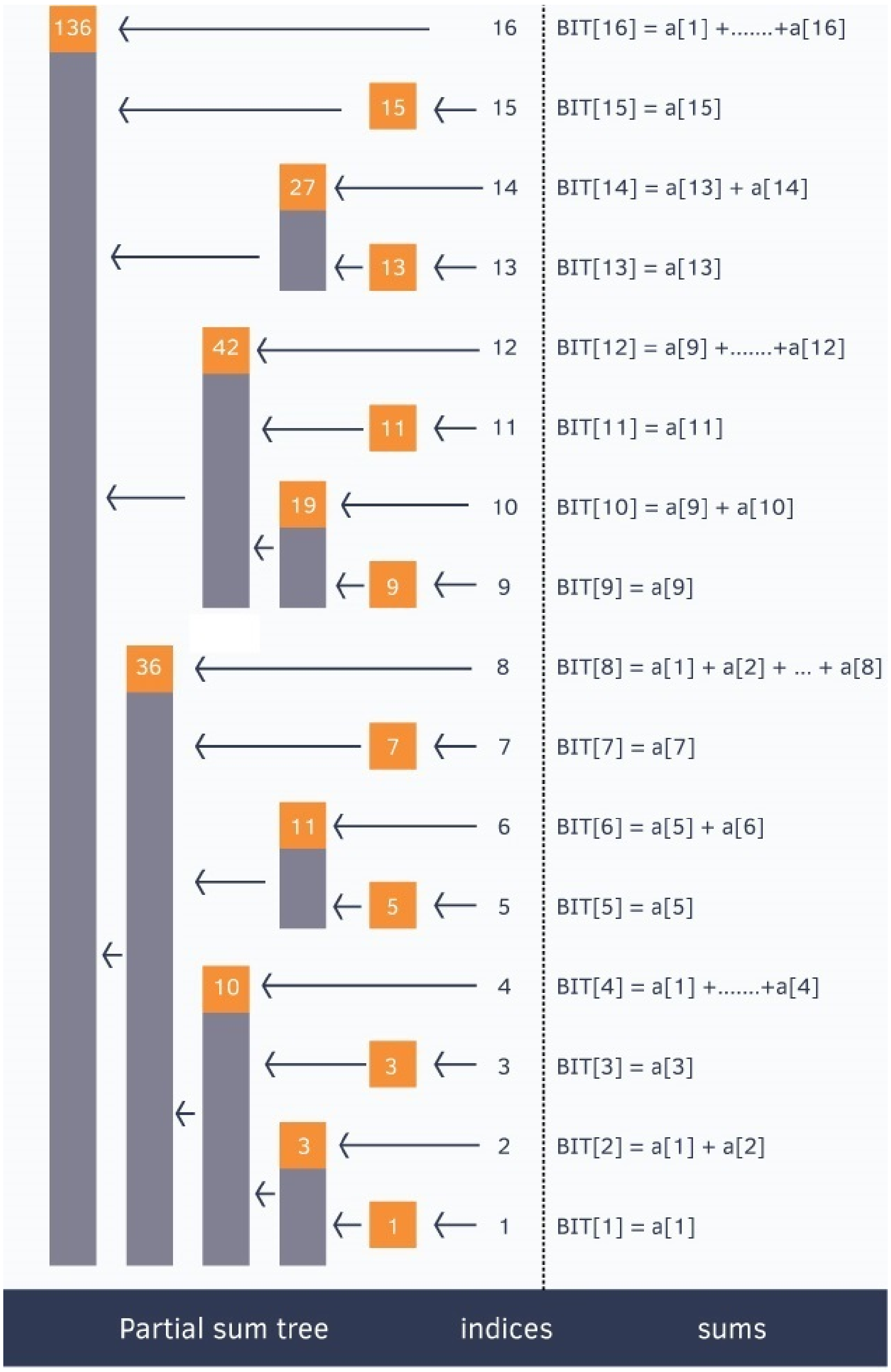
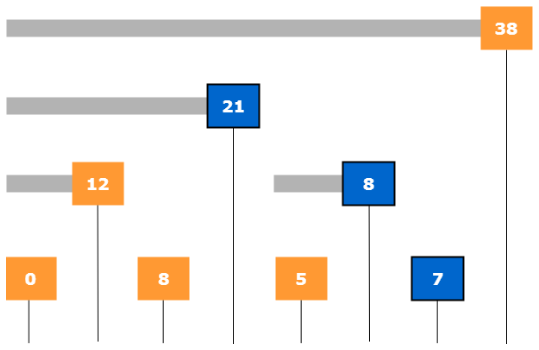


Figure 5.4: Example of a BIT for the array $a[] = \{$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16$\}$ [61]. For ease $a[0]$ is only used to make the array one-indexed, like BITs are, thus allowing a one-to-one mapping. Every index m in the $BIT[]$ array stores the cumulative sum from the index $m - (2^u) + 1$ to m (both inclusive), where u represents the last set bit in the index m (e.g. the last bit set in 6 is 2, as the binary representation is 110 and the last bit represents 2^1). The value in the enclosed box represents $BIT[\text{index}]$ and the grey bars indicate which elements are included in the partial sum over $a[]$ in each value of $BIT[]$.



$BIT[]$	$\{$0,	0,	12,	8,	21,	5,	8,	7,	38$\}$
Index/identity (i)		1	2	3	4	5	6	7	8
$a[i] = c_i v_i$	0	0	12	8	1	5	3	7	2
σ_i^{vin}	0	0	12	20	21	26	29	36	38

Figure 5.5: Calculating the prefix sum $\sigma_i^{vin} = \sum_{j=1}^{i} c_i v_i$ can be done efficiently by using BITs. For simplicity we assume that these values are saved in order of identity in this example. When we have a BIT which is mapped to the array $a[] = \{0, c_1v_1, ..., c_Kv_K\} = \{0, 0, 12, 8, 1, 5, 3, 7, 2\}$, the corresponding $BIT[]$ array contains the values $\{0, 0, 12, 8, 21, 5, 8, 7, 38\}$. Each node of the BIT with index i contains the value of $c_jv_j + c_{j+1}v_{j+1} + ... + c_iv_i$, where j is the index of its left-child, e.g. $BIT[4] = c_2v_2 + c_3v_3 + c_4v_4 = 21$. This is also indicated by the grey bars and has been shown previously in Figure 5.4. This means the prefix sum for an index i can be calculated by summing the BIT node with index i with each node it is in the right subtree of. As an example, the nodes that are summed when calculating prefix sum for index 7 are highlighted in blue, giving a total sum of 36.

Using these structures we can efficiently sample and update the BIT. We will show this using the BIT that keeps track of the prefix sums

$$\sigma_j^{vinyl} = \sum_{i=1}^{j} c_i v_i$$

with $1 \leq j \leq K$, as an example. The other prefix sums will not be shown, but will be tracked by BITs similarly. To track the sums for σ_j^{vinyl} we would need to create a BIT for the array $\{c_1r_1, c_2r_2, ..., c_Kr_K\}$, but, as indicated, we do not track the reactivity of species S_i for which $c_i = 0$. Instead, we create a BIT for the array $a[] = \{0, a_1, a_2, ..., a_{b-1}\}$ where

$$a_m = \begin{cases} 0 & \text{if } m \in \mathcal{E} \\ c_{id(m)} v_{id(m)} & \text{otherwise} \end{cases}$$

Note that all prefix sums σ_j^{inter},σ_j^{intra}, σ_j^{corr} and σ_j^{vinyl}, as defined in Section 5.2.2, contain the factor c_i.

Initialization The initialization of the BIT is relatively straightforward. For simplicity we assume that we start with at least one molecule of each starting species S_1=I,S_2=M and S_3=C, and the add the mappings $(1, 2, 3) \mapsto (1, 2, 3)$ to $index$. From this we construct the initial arrays $a[] = \{0, c_1v_1, c_2v_2, c_3v_3, 0\}$ and $BIT[] = \{0, c_1v_1, c_1v_1 + c_2v_2, c_3v_3,_1 v_1 + c_2v_2 + c_3v_3\}$. Furthermore, we have $b = 5$ and $\overline{\mathcal{E}} = (4)$.

Sampling We will now show that even though we only track reactivities for each $S_j \in \mathcal{H}$ rather than for each $S_i \in \{S_1, ..., S_k\}$, and these reactivities are not kept in order of the molecule species identity

j, we sample with the same probabilities. Recall that sampling is done by generating a uniformly distributed random number x in the interval $[0,1)$ and calculating j for

$$\sigma_{j-1}^{vinyl} < x\sigma_K^{vinyl} < \sigma_j^{vinyl},$$

as shown previously in Equation 5.7. This gives a probability of

$$\frac{\sigma_j^{vinyl} - \sigma_{j-1}^{vinyl}}{\sigma_K^{vinyl}} = \frac{c_j v_j}{\sigma_K^{vinyl}}$$

for $j \in \{1, ..., K\}$ to be selected. Note that this depends only on the reactive value $c_j v_j$ that corresponds to the identity j, and the total sum of reactive values σ_K^{vinyl}. This probability is independent of any indices. We will now show that both the reactive values and the total sum remain unaltered.

Firstly, restricting our sampling space to reactive values $c_j v_j$ of species $S_j \in \mathcal{H}$ does not change the total sum. By definition

$$\sigma_K^{vinyl} = \sum_{i=1}^{K} c_i v_i = \sum_{i \in \{j | S_j \in \mathcal{H}\}} c_i v_i$$

because $c_i = 0$ when $S_i \notin \mathcal{H}$, meaning that $c_i v_i = 0$ as well. The probability of identity j of species $S_j \notin \mathcal{H}$ to be selected remains zero, as removing j from the sampling space altogether will give the same probability.

Secondly, saving the reactive values of species $S_j \in \mathcal{H}$ in the array $a[]$ also does not change the total sum. There is a unique m for each $S_j \in \mathcal{H}$ for which $a[m] = c_j v_j$, dictated by the use of the map id in the definition of $a[]$. Because all elements in $a[]$ for which no mapping exists in id are zero we can thus also conclude that

$$\sum_{i \in \{j | S_j \in \mathcal{H}\}} c_i v_i = \sum_{m=1}^{b-1} a_m,$$

thus the total sum remains the same.

Lastly, sampling from the array $a[]$ also does not change the probabilities. Sampling on $a[]$ is done by generating a uniformly distributed random number x in the interval $[0,1)$ and calculating m for

$$\sum_{n=1}^{m-1} a[n] < x \sum_{n=1}^{b-1} a_n < \sum_{n=1}^{m} a[n].$$

This means that m has a probability of

$$\frac{\sum_{n=1}^{m} a[n] - \sum_{n=1}^{m-1} a[n]}{\sum_{n=1}^{b-1} a_n} = \frac{a[m]}{\sum_{n=1}^{b-1} a_n}$$

to be selected. If there is a mapping $m = index(j)$, then $a[m] = c_j v_j$ by definition, and the probability is

$$\frac{a[m]}{\sum_{n=1}^{b-1} a_n} = \frac{c_j v_j}{\sigma_K^{vinyl}}.$$

Converting this index m to identity j using the inverse mapping $j = id(m)$ results in the same probability of j to be selected. If there is no mapping $m = index(j)$, then $m \in \mathcal{E}$ and $a[m] = 0$, meaning there is a probability of zero for m to be selected.

Performing sampling is thus relatively straightforward. We obtain m for

$$\sum_{n=1}^{m-1} a[n] < x \sum_{n=1}^{b-1} a_n < \sum_{n=1}^{m} a[n]$$

using $m = getIndex(xBIT[b-1], b-1)$. Subsequently, we obtain the identity j of the respective molecule species using $j = id(m)$.

Updating When simulating a reaction the numbers of molecules of the reactant species and the product species are altered. Updating the BIT to reflect this is done in the following steps:

- For each reactant species S_i in the simulated reaction the number of molecules of that species c_i is decreased by one. This means that the vinyl reactivity of the species changes from $c_i v_i$ to $(c_i - 1)v_i = c_i v_i - v_i$. We decrease the reactivity saved in $a[]$ by performing the update $a[index(i)] := a[index(i)] - v_i$ for each reactant species S_i. Since the array the BIT is mapped to is altered, $BIT[]$ needs to be updated as well. This is done by performing the using the function $adj(index(i), -v_i)$.
- The number of molecules c_k of the product species S_k of the simulated reaction is increased by one. If there exists a mapping for k in $index$, i.e. S_k is a non-empty species, updating $a[]$ and $BIT[]$ is done by performing $a[index(k)] := a[index(k)] + v_k$ and $adj(index(k), v_k)$ respectively.
- If the product species S_k is an empty species, i.e. $S_k \notin \mathcal{H}$, we perform insertion, which we will now describe.

Insertion Insertion of the reactivity $c_k v_k$ of S_k is done by finding the smallest unused index in the BIT, i.e. $m_1 \in \overline{\mathcal{E}} = (m_1, ..., m_{|\mathcal{E}|})$, and updating the arrays $a[]$ and $BIT[]$ by $a[m_1] := c_k v_k$ and $adj(m_1, c_k v_k)$. The entry $k \mapsto m_1$ is added to our $index$ mapping, which removes m_1 from $\mathcal{E}$. However, if we try to perform insertion when $\mathcal{E} = \emptyset$ we run into a problem, as this is no longer possible.

At that point we double the arrays to allow for further insertions. We copy the entire arrays to one of size $2b-1$ before insertion is performed by $a[] := \{0, a_1, a_2, ..., a_{2b-1}\}$ with

$$a_m = \begin{cases} a[m] & \text{if } 0 < m < b \\ 0 & \text{otherwise} \end{cases}$$

and $BIT[] := \{0, a_1, a_2, ..., a_{2b-1}\}$ with

$$a_m = \begin{cases} BIT[m] & \text{if } 0 < m < b \\ BIT[b-1] & \text{if } m = 2b-1 \\ 0 & \text{otherwise} \end{cases}$$

Note that we copy the total sum to the last element in $BIT[]$. Subsequently we update b by performing $b := 2b - 1$. Doubling the array introduces new unused indices, due to the absence of a mapping to these indices in $index$. This means that insertion has a worst-case time complexity of $\mathcal{O}(b)$, but a far lower average-case time complexity as the arrays only need to be doubled in size sporadically.

Deletion When simulating a reaction the numbers of molecules of the reactant species are altered. If, after performing the simulation the number of molecules c_i of one of the reactant species is zero, we remove this species from $a[]$ and $BIT[]$. We do this simply by removing $index(i) \mapsto m$ from our index mapping, adding m to the set of unused indices $\mathcal{E}$. This allows the index to be reused for insertion, thus preventing unnecessary growth of the tree.

Since we want to keep our BIT as small as possible we do not only prevent unnecessary growth, but also shrink the tree when possible. When $|\mathcal{E}| \geq (b-1)/2$, i.e. half or more of the indices in the tree are unused, the BIT is halved.

- The first step in this process is the relocation of all elements in the second half of $a[]$ to the unused indices in the first half, and updating $BIT[]$ accordingly. For each used index m in the second half of the tree, i.e. $m > (b-1)/2 \wedge m \notin \mathcal{E}$, we:
  - obtain $m_1 \in \overline{\mathcal{E}} = (m_1, ..., m_{|\mathcal{E}|})$
  - update $a[m_1]$ and $BIT[m_1]$ by performing $a[m_1] := a[m]$ and $adj(m_1, a[m])$
  - update $a[m]$ and $BIT[m]$ by performing $adj(m, -a[m])$ and $a[m] := 0$
  - update the mapping $index(j) \mapsto m$ to $index(j) \mapsto m_1$, thus also removing m_1 from $\mathcal{E}$
- The second step is halving the arrays, such that $a[] := \{0, a[1], ..., a[(b-1)/2]\}$ and $BIT[] := \{\{0, BIT[1], ..., BIT[(b-1)/2]\}\}$, and updating b by $b := (b-1)/2$.

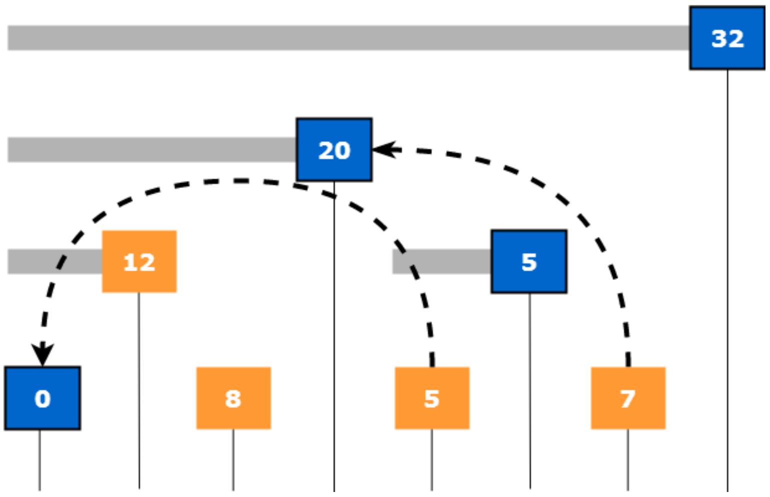
BIT halving is thus a very costly operation and has a worst-case time complexity of $\mathcal{O}(b\ log(b))$. The trade-off for this deletion strategy is that a lower value for b also means less time is needed to perform updates and binary searches, which take up most of the operations on the BIT. To give a better average-case time complexity we prioritize filling the unused indices in the left half of the tree first, due to the use of $\overline{\mathcal{E}}$ rather than $\mathcal{E}$ when performing insertions. Furthermore, to prevent continuous doubling and halving the BIT we implemented a threshold so a BIT has to be half full minus a fixed number of entries. It is important to note that BIT halving is mostly performed as the number of non-empty species decreases, which in our case is after gelation. Halving is thus especially used to speed up post-gelation simulation.

5.4 Structural model

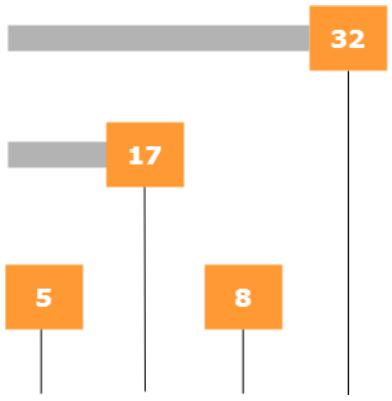
Requirement R 3, as given in Section 1.4, states that one of the possible results the simulation has to be able to yield is a 3D model of the structure of a polymer in the simulation. For the model of molecular structures we use graphs similar to the schematic representation we use in this work, like the one in Figure 5.3. We use graphs to represent polymer networks, as graphs are a natural way to represent networks. In this section we will first give a definition of the graphs that are used. Subsequently we will give the requirements which the graph encoding has to fulfill. Finally, we show how these graphs are encoded into a structural model in such a way that these requirements are fulfilled.

5.4.1 Graphs

The specific type of graph that is used to represent polymer structures is a labeled directed graph. The node labels are used to indicate the different types of molecule the nodes represents. We use ‘I’ for half-initiators, ‘M’ for monomers, and ‘C1’ and ‘C2’ for the first and second half of the crosslinker. In this case ‘first’ refers to the order in which the halves of the crosslinker are connected to a polymer. Edges in the graph are directed, starting at the node representing the molecule that contained the radical and ending at the node representing the molecule that contained the vinyl group



$BIT[]$	{0,	0,	12,	8,	20,	5,	5,	7,	32}
$a[]$	{0,	0,	12,	8,	0,	5,	0,	7,	0 }
Index (m)	0	1	2	3	4	5	6	7	8
$m \in \mathcal{E}$	$false$	$true$	$false$	$false$	$true$	$false$	$true$	$false$	$true$
Identity (i)	-	-	2	3	-	5	-	7	-
$c_i v_i$	-	-	12	8	-	5	-	7	-



$BIT[]$	{0,	5,	17,	8,	32}
$a[]$	{0,	5,	12,	8,	7 }
Index (m)	0	1	2	3	4
$m \in \mathcal{E}$	$false$	$false$	$false$	$false$	$false$
Identity (i)		5	2	3	7
$c_i v_i$		5	12	8	7

Figure 5.6: Halving the BIT is done by moving values in the second half of $a[]$ to the first half and updating the BIT accordingly, as described by the algorithm in Section 5.3.1.
Top: Example of a BIT that is half empty. The arrows indicate the moving of elements in $a[]$ and the highlighted blue nodes indicate that the index m of this node is unused. We start with the index map $index : (2, 3, 5, 7) \mapsto (2, 3, 5, 7)$.
Bottom: resulting BIT after moving elements from the second half of $a[]$ to the first half. Note that the order of identities has also changed, as the mapping between indices and identities has been updated as well. The resulting index map is $index : (1, 2, 3, 4) \mapsto (5, 2, 3, 7)$.

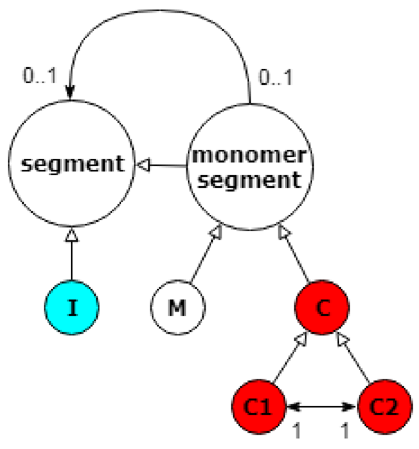


Figure 5.7: Type graph for the graphs used to represent polymer structures.

at the moment they became connected. The exception to this are the bidirectional arrows between the nodes representing the halves of a crosslinkers These arrows indicate that the two connected half-crosslinkers nodes are part of the same crosslinker. The type graph for our graph model is given in Figure 5.7. The reactive groups are not explicitly indicated in the graphs, but follow from the graph structure as follows:

- Reactive centers are located on nodes with outgoing unidirectional arrows, but no incoming unidirectional arrows. We have given these nodes a bolded solid outline for clarity. Additional reactive centers are located on I-nodes with no incoming (unidirectional) arrows. Each of the nodes representing molecules with reactive centers have an associated position Z. Reactive centers on molecules represented by I-nodes and M-nodes are in position EN, those represented by C1-nodes are in position EC and those represented by C2-nodes in position MC.

- Pendent vinyl groups are located on C1-nodes and C2-nodes with no outgoing unidirectional arrows. We have given these nodes a bolded dotted edge for clarity.

An example of this graph model is shown in Figure 5.8.

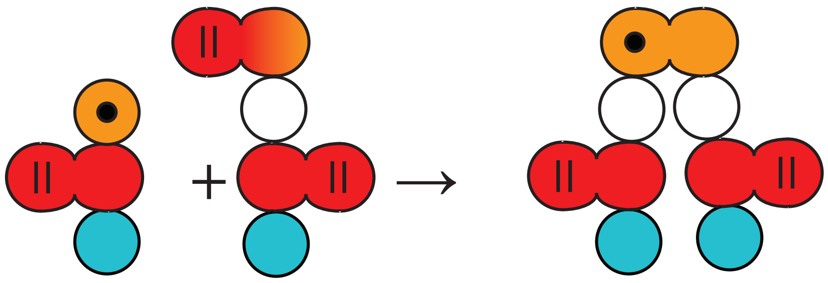
5.4.2 Requirements

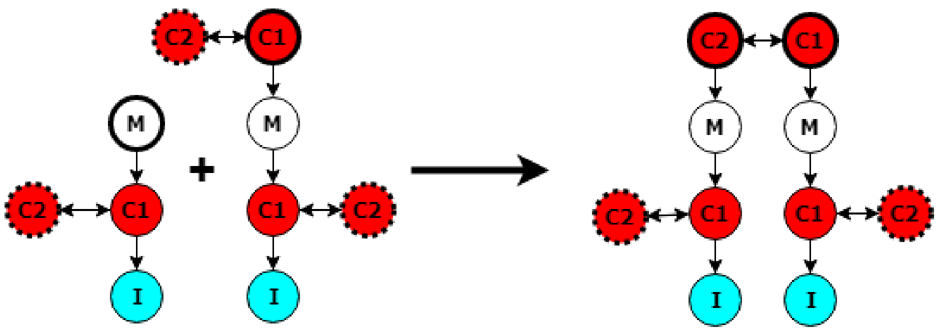
The graphs are encoded into what we refer to as the structural model. The structural model $\overline{S_i}$ for molecule species S_i has been mentioned previously in Section 5.2.4 and is used multiple times during the simulation:

- It is read in its entirety only once, when converting it to a graph file format that is supported by other programs. This is only done for a select number of structural models.

- It is created only once and is in use for as long as the molecule species exists in the simulation. It is not updated as it does not need to be changed.

- When simulating a reaction which yields a species S_k for which no structure exists, $\overline{S_k}$ is created by combining the structures of the reactant species which yielded S_k. This is done by connecting a node representing a molecule with a reactive center in the structure of the radical species to a node representing a molecule with a vinyl group in the structure of the vinyl species. There are multiple candidates for these nodes. We must thus be able to select specific nodes from the


(a) Schematic notation



(b) Graph model

Figure 5.8: Example of crosslinking reaction in both the schematic notation of Chapter 1 and the equivalent graph models. The RAFT agent that is attached to the reactive center in the vinyl molecule is present, but not shown in the schematic notation.

list of nodes representing molecules with reactive groups in the structural model. This selection process is done using random sampling.

A species is discovered only once, but can undergo many reactions after its discovery. Sampling is thus the most often occurring operation, followed by creation and reading. To reflect this we have chosen to focus on a structure that can:

- be sampled for a node representing a molecule with a reactive group in $\mathcal{O}(1)$ time.
- be created in $\mathcal{O}(n)$ time, where n is the number of reactive groups.
- be converted to a 3D model in $\mathcal{O}(m)$ time, with m being the number of molecules that make up the polymer to which the structural model belongs.

Furthermore, to keep memory usage down, we want to describe molecule structures as the combination of the structures that were combined to create it. We want to have a worst-case memory complexity of $\mathcal{O}(n)$ time, where n is the number of reactive groups. The structural model we will now describe fulfills these requirements.

5.4.3 Encoding

Encoding of the graph model into the structural model is done in such a way that the polymers are not treated as network structures, but rather as several interconnected polymer chains. These polymer chains are represented as sequences of node labels. We first define our set of node labels

$$\mathcal{T} = \{\overline{I}, \overline{M}, \overline{C_1}, \overline{C_2}\}. \tag{5.9}$$

These node labels are also used in our graph model, as described in Section 5.4.1, but given an overline to be able to distinguish them from the molecule species I and M. Subsequently, we define the set of possible sequences

$$\mathcal{N} = (\overline{M} \vee \overline{C_1} \vee \overline{C_2})^* \overline{I}, \tag{5.10}$$

which are essentially all possible sequences of labels that end at the first occurrence of the label $\overline{I}$. These sequences are used to encode the sequences of nodes connected via unidirectional arrows in our graph model. As such, these sequences can not only be used to encode chains of nodes, but also nodes within these chains. For example, we can use the sequence $(\overline{M}, \overline{I})$ to represent the monomer closest to half-initiator in the chain $(\overline{M}, \overline{C_1}, \overline{M}, \overline{I})$. A chain represented by the sequence n thus contains the nodes

$$nodes(n \in \mathcal{N}) \mapsto \begin{cases} \{\overline{I}\} & \text{if } n = \overline{I} \\ \{(t||n')\} \cup nodes(n') & \text{if } n = (t \in \mathcal{T}||n' \in \mathcal{N}) \end{cases}$$

where '$||$' is the concatenation operator, which combines sequences, i.e. $(\overline{M}||(\overline{M}, \overline{I})) = (\overline{M}, \overline{M}, \overline{I})$. We also define the function

$$type((t_1, t_2, ..., \overline{I}) \in \mathcal{N}) \mapsto t_1 \tag{5.11}$$

which gives us the label of the first element in the sequence. This function can be used to obtain the type of a node using the sequence that encodes it.

One of the limitations of encoding a chain of nodes in a sequence is that the resulting sequences may fail to be unique. In the graph network there may be multiple chains with the same order of node labels. The sequences cannot be used to identify a unique chain in the network in that case. To solve this, each chain is given a unique identifier $m \in \mathbb{Z}^+$. Using the pair (n, m) where $n \in \mathcal{N}$, we can now also uniquely identify a node in our structural model.

Such pairs are also used to describe the possible set of crosslinks

$$\mathcal{L} = \{((n, m), (n', m')) | n, n' \in \mathcal{N}, m, m' \in \mathbb{Z}^+, type(n) = \overline{C_1}, type(n') = type\overline{C_2}\} \tag{5.12}$$

between nodes labeled $\overline{C_1}$ and $\overline{C_2}$, where m and m' are again unique chain identifiers. These crosslinks will be saved in trees, defined by 4-tuples of the set

$$O = \{\langle l, \alpha, o^{rad}, o^{vin} \rangle | o^{rad}, o^{vin} \in O, \alpha \in \mathbb{Z}^*, l \in \mathcal{L}\} \cup \{\varnothing\}, \tag{5.13}$$

where l is the link that links models of a radical and vinyl species together, o^{rad} and o^{vin} are the subtrees containing the crosslinks in the models of these radical and vinyl species, and α is the offset used to update chain indices in o^{vin} without altering o^{vin} itself. We show why the use of α is necessary further on in this section.

Structural model Using these structures we can now define the structural model $\overline{\mathcal{S}_i}$ of a molecule of species S_i by

$$\overline{\mathcal{S}_i} = \langle D_i, Y_i, o_i \rangle \tag{5.14}$$

with:

- The list of polymer chains that make up the polymer structure

$$D_i = (d_1, d_2, ..., d_{r_i}), d_j \in \mathcal{N}.$$

  This list is indexed, and the index is used to uniquely identify polymer chains. This list is also used as the list of reactive centers, as these are located on the chain-ends. The nodes $\{(d_j, j) | d_j \in D_i\}$ thus represent the molecules with a reactive center in the structure.

- The list of vinyl groups in the polymer structure

$$Y_i = ((n_1, m_1), (n_2, m_2), ..., (n_{v_i}, m_{v_i})), m_j \in \{1, 2, ..., r_i\}, n_j \in nodes(d_{m_j}), type(n_j) = \overline{C_1}.$$

  The pendent vinyl groups are located on molecules we represent with C_2-nodes. However, these nodes are not yet part of a polymer chain and thus have no respective node in the sequences of D_i. This is why we refer to these C_2-nodes by the C_1-nodes they are linked to with a bidirectional edge. This means that the C_2-nodes representing molecules with vinyl groups are implicit nodes.

- The tree $o_i \in O$ containing crosslinks in the structure of $\overline{S_i}$.

The numbers r_i and v_i are the number of reactive centers and vinyl groups in a molecule of species S_i, as defined in Section 4.3. One restriction of this graph encoding is that there are is no encoding for graphs of species M and C due to these species not having reactive centers. This is the reason we only track structures of polymers in our simulation.

If we encode the graphs in the reaction of Figure 5.8b with structure $S_i + S_j \to S_k$ we get the structural models

$$\overline{S_i} = \langle [(\overline{M}, \overline{C_1}, \overline{I})], [((\overline{C_1}, \overline{I}), 1)], \varnothing \rangle, \tag{5.15}$$

$$\overline{S_j} = \langle [(\overline{C_1}, \overline{M}, \overline{C_1}, \overline{I})], [((\overline{C_1}, \overline{I}), 1), ((\overline{C_1}, \overline{M}, \overline{C_1}, \overline{I}), 1)], \varnothing \rangle \tag{5.16}$$

and

$$\begin{aligned} \overline{S_k} = &\langle [(\overline{C_2}, \overline{M}, \overline{C_1}, \overline{I}), (\overline{C_1}, \overline{M}, \overline{C_1}, \overline{I})], \\ &[((\overline{C_1}, \overline{I}), 1), ((\overline{C_1}, \overline{I}), 2)], \\ &\langle (((\overline{C_1}, \overline{M}, \overline{C_1}, \overline{I}), 2), ((\overline{C_2}, \overline{M}, \overline{C_1}, \overline{I}), 1)), 1, \varnothing, \varnothing \rangle \rangle. \end{aligned} \tag{5.17}$$

We can see clearly in S_k why the index m_j in the definition of $(n_j, m_j) \in Y_k$ is necessary, as $n_1 = n_2$, but these sequences are not used to refer to the same pendent vinyl group. Furthermore, note that S_k could also be encoded by other tuples where the chains are numbered in another order. However, as we have thus far kept radical molecules on the left and vinyl molecules on the right in every notation, we do the same when creating D_k and Y_k in the structure of the product species.

Reaction simulation with structural models In Section 5.2.4 we have show that when a reaction which yields the product species S_k is simulated, and no structure $\overline{S_k}$ exists for this species, a new structure is created. If the simulated reaction is an intermolecular reaction $S_i + S_j \to S_k$, we combine $\overline{S_i}$ and $\overline{S_j}$ by creating an edge between a node representing a molecule with a reactive center in the former and a node representing a molecule with a vinyl group in the latter. If the reaction is an intramolecular reaction $S_i \to S_k$, we create $\overline{S_k}$ by creating an edge between a node representing a molecule with a reactive center and one representing a molecule with a vinyl group in $\overline{\mathcal{S}_i}$. There possibly are multiple combinations of such nodes that can be connected, so a selection has to be made. The specific nodes are selected as follows:

- In Section 5.4.1 we mentioned that there is a reactive group position Z associated with each node label $t \in \mathcal{T}$. Using the selection process for Z explained in Section 5.2.2 we would thus need to also find a node representing a molecule with a reactive center that has a node label of the correct type for Z. An easy way to obtain a node with a label corresponding to position Z is to obtain Z itself by sampling the structure, rather than using the selection procedure described in Section 5.2.2. A uniformly distributed random integer p in the interval $[1, r_i]$ is generated, and the node associated with the sequence $d_p \in D_i$ is selected. Depending on the node label we obtain the following value for Z:

$$Z = \begin{cases} EN & \text{if } type(d_p) = \overline{M} \\ EC & \text{if } type(d_p) = \overline{C_1} \\ EM & \text{if } type(d_p) = \overline{C_2} \end{cases} \tag{5.18}$$

  Additionally, the selected node d_p is used as the node representing the molecule with a reactive center that is going to react. We will refer to this node by its index p.

- The node representing a molecule with a vinyl group is selected similarly. In the case of an intramolecular reaction, or an intermolecular reaction with a vinyl polymer, we generate a uniformly distributed random integer p' in the interval $[1, v_i]$, and the implicit C_2-node associated with $y_{p'} \in Y_i$ or $_{p'} \in Y_j$ is selected respectively. We will refer to this C_2-node using the index p'. In the case of an intermolecular reaction with a crosslinker or monomer, we simply use $p' = 1$.

Using the structures $\overline{\mathcal{S}_i}$ and $\overline{\mathcal{S}_j}$ of the selected reactant species in combination with the indices p and p' we can create the structure $\overline{\mathcal{S}_k}$ of the product species using the function $product(\overline{\mathcal{S}_i}, \overline{\mathcal{S}_j}, p, p')$ for intermolecular reactions and $product(\overline{\mathcal{S}_i}, p, p')$ for intramolecular reactions, as defined by Equations 5.19 and 5.20. We would like to point out multiple aspects of these functions:

- The concatenation operator '||' is used here to concatenate lists as well.
- The references to nodes obtained from the structure $\overline{\mathcal{S}_j}$ in the $product$ function for intermolecular reactions need to be updated when S_j is a polymer species. Due to the concatenation of D_i and D_j the references (n, m) in both Y_j and o_j or off by $|D_i| = r_i$. This is why every (n, m) in Y_j is replaced by $(n, m + r_i)$ when creating Y_k. For o_j a different approach is taken: rather than altering the tree o_j itself when creating o_k, o_k contains both a reference to o_j and the offset $\alpha = r_i$ caused by the chain list concatenation. This offset is used to correctly calculate the chain indices when converting the structural to a graph, as shown in Equations 5.21 and 5.22.
- Explicit C_2-nodes are created when the vinyl species S_j is a polymer, i.e. a reaction between a radical and a pendent vinyl group occurs. These nodes *are* part of a chain.

$$product(\overline{\mathcal{S}_i}, \overline{\mathcal{S}_j}, p, p') \mapsto \begin{cases} \langle [(\overline{M}, \overline{I})], \emptyset, \varnothing \rangle & \text{if } S_i = I \wedge S_j = M \\ \langle [(\overline{C_1}, \overline{I})], [(\overline{C_1}, \overline{I}), 1], \varnothing \rangle & \text{if } S_i = I \wedge S_j = C \\ \langle [(\overline{C_2}, \overline{I})] \mid\mid D_j, [(n_x, m_x + 1) | (n_x, m_x) \in Y_j, x \neq p'], \langle ((n_{p'}, m_{p'} + 1), ((\overline{C_2}, \overline{I}), 1)), 1, \varnothing, o_j \rangle \rangle & \text{if } S_i = I \wedge S_j = P_j \\ \langle [d_1, ..., d_{p-1}, (\overline{M} || d_p), d_{p+1}, .., d_{r_i}], Y_i, o_i \rangle & \text{if } S_i = P_i \wedge S_j = M \\ \langle [d_1, ..., d_{p-1}, (\overline{C_1} || d_p), d_{p+1}, .., d_{r_i}], Y_i \mid\mid [((\overline{C_1} || d_p), p)], o_i \rangle & \text{if } S_i = P_i \wedge S_j = C \\ \langle [d_1, ..., d_{p-1}, (\overline{C_2} || d_p), d_{p+1}, .., d_{r_i}] \mid\mid D_j, Y_i \mid\mid [(n_x, m_x + r_i) | (n_x, m_x) \in Y_j, x \neq p'], & \text{if } S_i = P_i \wedge S_j = P_j \\ \qquad \langle ((n_{p'}, m_{p'} + r_i), (\overline{C_2} || d_p)), r_i, o_i, o_j \rangle \rangle & \end{cases} \tag{5.19}$$

$$product(\overline{\mathcal{S}_i}, p, p') \mapsto \langle [d_1, ..., d_{p-1}, (\overline{C_2} || d_p), d_{p+1}, .., d_{r_i}], [y_1, ..., y_{p'-1}, y_{p'+1}, ..., y_{v_i}], \langle ((n_{p'}, m_{p'}), (\overline{C_2} || d_p)), 0, o_i, \varnothing \rangle \rangle \tag{5.20}$$

$$links(o \in O) \mapsto links(o \in O, 0) \tag{5.21}$$

$$links(o \in O, x \in \mathbb{Z}^*) \mapsto \begin{cases} \emptyset & \text{if } o = \varnothing \\ \{((n, m + x), (n', m' + x))\} \cup link(o^{rad}, x) \cup link(o^{vin}, x + \alpha) & \text{if } o = \langle ((n, m), (n', m')), \alpha, o^{rad}, o^{vin} \rangle \end{cases} \tag{5.22}$$

Conversion to graph To be able to convert the nodes in a structure $\overline{\mathcal{S}_i}$ back to nodes of its respective graph G_i we must solve the following problem. Every molecule in a polymer chain is represented by a pair $(n \in \mathcal{N}, m \in \mathbb{Z}^+)$ in the encoded structure. However, half-crosslinkers with a pendent vinyl group are not part of a chain, thus do not have a representative pair in our encoded structure. We have shown previously that these half-crosslinkers are represented by implicit C_2-nodes, attached to referenced C_1-nodes in Y_i that represent the other half of the crosslinker.

To create a unique representation for these C_2-nodes we have extended our pair to a triple $(n \in \mathcal{N}, m \in \mathbb{Z}^+, b \in \{true, false\})$ by adding the boolean b, which indicates whether the triple represents an implicit C_2-node. This triple will be used as the node identity in G_i.

We can now define G_i using the $nodes$ and $links$ functions as

$$G_i = \langle V_i, E_i \rangle \tag{5.23}$$

with the list of vectors

$$V_i = \{(n, m, false) | n \in nodes(d_m), d_m \in D_i\} \cup \{(n, m, true) | (n, m) \in Y_i\}, \tag{5.24}$$

the list of edges

$$\begin{aligned} E_i = &\{((n, m, false)|n', m, false))|n = (t \in \mathcal{T}||n' \in \mathcal{N}) \in nodes(d_m), d_m \in D_i\} \\ &\cup \{((n, m, \neg b), (n, m, b))|(n, m) \in Y_i, b \in \{true, false\}\} \\ &\cup \{((n, m, false), (n', m', false))|((n, m), (n', m')) \in links(o_i)\} \\ &\cup \{((n, m, false), (n', m', false))|((n', m'), (n, m)) \in links(o_i)\} \end{aligned} \tag{5.25}$$

and the label function

$$label((n, m, b) \in V_i) \mapsto \begin{cases} \overline{C_2} & \text{if } b = true \\ label(n) & \text{otherwise} \end{cases} \tag{5.26}$$

Chapter 6

Implementation

The algorithm discussed in the previous chapter has been implemented in Java 8. Because the simulator needs to be dynamic it is not sufficient to only implement the algorithm; the simulator should also be configurable without changing the implementation. This applies to both input and output configuration. Input parameters should be variable to allow for the simulation of multiple experiments. The output processing should be variable so we can extract dynamic properties from the simulation.

In this chapter we discuss the implementation of our simulator. We will first discuss the general structure of our simulator in Section 6.1. Subsequently, in Section 6.2, we discuss the configuration of the simulation and how this is used to create the initial state. Lastly, in Section 6.3, we discuss the dynamic output of the simulator.

6.1 Simulation structure

The program structure is relatively straightforward. An Excel file contains the program's configuration and functions as the input. This file is parsed by the simulator before starting the simulation and simulation parameters are extracted. These simulation parameters are then used to create the initial state of the simulation.

Subsequently the simulation follows the algorithm described in the previous chapter. A random reaction is selected, this reaction is either rejected and a new one is chosen, or is accepted. Once a reaction is accepted it is simulated and the simulation state is updated accordingly.

The halting criteria are checked after each simulated reaction and the simulation either continues or stops. At this point the conversion is also checked, since output is given at a fixed conversion interval. Some basic information is printed to the command line interface, including conversion, number of molecule species and the size of the biggest polymer in the simulation. Output data is extracted from the simulation state as well. This output data will be further discussed in Section 6.3. If the simulation also includes molecules structures 3D models may also be generated at this point in the simulation. The 3D models will be further discussed in Section 6.3.

Once the simulation stops, the output data is converted to comma separated values (CSV) files that can be read by Excel. The program flow that was just described is also illustrated in Figure 6.1.

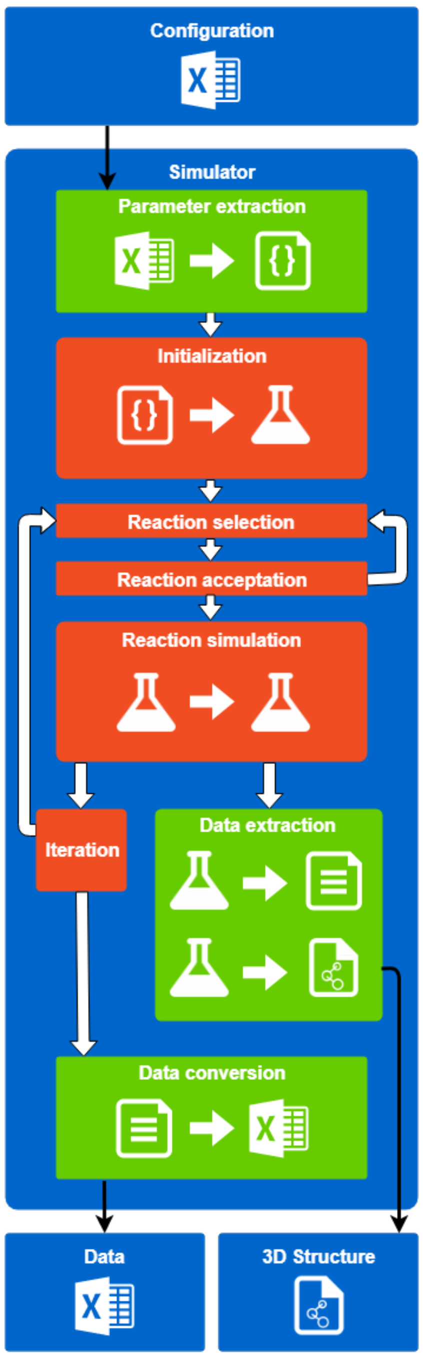


Figure 6.1: Program structure of our simulation. White arrows indicate program flow, black arrows indicate I/O. The separate files are indicated by the blue squares, internal I/O by green squares and the core algorithm in the orange squares.

Table 6.1: Simulation configuration parameters

Parameter	**Description**
	Needed by simulation algorithm
[I]	Concentration of initiator radicals (M)
[M]	Concentration of (monovinyl) monomer (M)
[C]	Concentration of crosslinker (M)
[RAFT]	Concentration of RAFT agent (M)
c_T	Total number of molecules in simulation
ϕ	Steric hindrance factor for reaction between mid-chain radical and monomer/crosslinker
ψ	Steric hindrance factor for reaction between end-chain radical and pendent vinyl group
ω	Steric hindrance factor for reaction between mid-chain radical and pendent vinyl group
vol_i	Function giving interaction volume of molecule species i
l	Segment length of a single monomer (nm); can be used in v_i
	Needed for simulation with molecule structures
I_{atom}	2-letter code of atom representing half-initiator in 3D structure
M_{atom}	2-letter code of atom representing monomer in 3D structure
C_{atom}	2-letter code of atom representing half-crosslinker in 3D structure
P_{atom}	2-letter code of atom representing half-crosslinker with pendent vinyl group in 3D structure
$\mathcal{X}$	Set of comma-separated 3D model sizes; functions as maximums, as most 3D tools have upper limits for number of atoms in 3D models.
d	Detail level of simulation, either low or high
	Needed for output
$\mathcal{F}$	Set of comma-separated (custom) output functions
w_I	Weight of half-initiator (Da)
w_M	Weight of monovinyl monomer (Da)
w_C	Weight of crosslinker (Da)

Table 6.2: Excel configuration generated from Java class shown in Figure 6.2.

Length of monomer (nm)	0.3
Number of molecules	1,000,000,000
Concentration of initiator radicals (M)	0.0035
Polymer interaction volume equation	pi*4/3(l*sqrt((l+M+2C-V)/6))ˆ3
...	...

6.2 Configuration

The list of input parameters in the configuration file is shown in Table 6.1. In addition to the parameters needed by the simulation algorithm, the configuration also contains parameters for tweaking the output of the simulation.

The biggest addition to this configuration is a set of comma-separated output functions, which are used to extract output data from the simulation dynamically. These functions are explained in detail in Section 6.3. Another addition are the weights of the molecules that make up a polymer. With these weight the total weight of a polymer can be calculated, which is an essential step in creating a molecular weight distribution.

Several parameters are used by the *structured simulation*, which has the creation of 3D models of polymers as its goal; different 2-letter atom codes can be used to indicate a node in the structure of the 3D model. These are necessary to switch node colors. Since different 3D model viewers for molecule structures do not necessarily use the same colors for atoms. If these colors would not be changeable, the user could possibly be unable to distinguish different types of nodes. Furthermore a list of maximum model sizes is given, which will be further explained in Section 6.3.

All parameters are defined in the Excel configuration, which has a straightforward structure. The first column contains parameter descriptions and the second column contains the values entered by the user for these parameters.

6.2.1 Parameter extraction

The simulation parameters listed in the previous section have to be parsed in such a way that our simulation is able to use them. For this purpose we have created a `Settings` class in Java containing variables for all the simulation parameters. The parameters are defined as `public static` variables, meaning they can be accessed where necessary throughout the entire simulation program without needing access to an instance of the Settings class.

The goal of the parameter extraction is thus to read the Excel data and write the entered values to the associated Java variables. To fulfill this goal a mapping has to be created from the Excel data to the Java variables. Rather than using the Java variables names, which can be somewhat confusing for users, we use a custom `@Input` annotation to map Excel rows to Java variables. This annotation takes a String containing the description of the variable as an argument, which is then also used in the Excel file, essentially linking the two together. To prevent unlinking of the variable and the Excel row, we have locked the Excel column containing the variable descriptions. When reading values in the Excel file we are able to override the static variables in our `Settings` class. This should be done as the first step in the simulation process, to prevent usage of the variables before overriding.

An example of a possible mapping: The Excel table in Table 6.2 can be mapped to the `Settings` class shown in Figure 6.2. Note that only annotated variables have to be present in the Excel configuration.

```
public class Settings{

  public static final double AVOGADRO = 602214085700000000000000d;

  @Input("Length of monomer (nm)")
  public static double l = 0.3d;

  @Input("Number of molecules")
  public static long NR_OF_MOLECULES = 1000*1000*1000;

  @Input("Concentration of initiator radicals (M)")
  public static double INITIATOR_CONCENTRATION = 0.0035d;

  @Input("Polymer interaction volume equation")
  public static String VOLUME_EQ = "pi*4/3(l*sqrt((l+M+2C-V)/6))^3";

  ...

}
```

Figure 6.2: Extract from Java Settings class

6.2.2 Excel generation

Since there is a one-to-one mapping from annotated Java variables to Excel rows,we cannot only use these annotations to *read* the configuration file, but to generate it as well. When the simulator is started and no configuration file is detected, the program generates the configuration file, containing an Excel row for each annotated variable in the `Settings` class.

The Java-Excel converter currently supports native Java classes (byte, short, int, long, float, double, boolean, char), their class equivalents, String, BigInteger, BigDecimal, Enum and Java Interfaces. Numeric Java classes are converted to numeric Excel cells and characters and strings to text cells. Booleans, enumerations and Java interfaces are converted to drop lists. The drop list for booleans have 'TRUE' and 'FALSE' as option, enumerations have all possible values as options and Java interfaces have all implementations with a public constructor without arguments as options.

This dynamic Java-Excel converter allows easy additions to our Java settings class, as well as removal from the Excel file. E.g. if we would want to add Avogadro's number to our Excel configuration in Figure 6.2, we would only have to remove the `final` modifier and add an `@Input("Avogadro's number")` annotation. Furthermore, if we would decide that the `VOLUME_EQ` variable should not be changeable anymore, we would only need to remove the annotation, at which point the simulation will just fall back to the default value of the static variable Changing the input of the simulation is as easy as changing the annotations in the settings class and regenerating the Excel file.

6.2.3 Parsing of mathematical equations

One of the parameters of our simulation is vol_i, the equation for the interaction volume of polymers, which is entered in the Excel configuration as a string. Performance for calculating the volume is important, since this has to be computed a large number of times during the simulation. We have compared multiple libraries that interpret the equation, but none gave us the performance we aimed for. Instead we have opted for another solution: conversion to Java code and runtime compiling of this code. This gives us similar performance to Java with the only extra overhead being the one-time conversion of the string representation of the equation to an instance of a Java class.

For the conversion from string to Java code we have used a simple ANother Tool for Language

Table 6.3: Expression elements

Element	Converted to
`pi`, π	Math.pi
`sqrt`, $\sqrt{}$	Math.sqrt()
ˆ	Math.pow()
$*, /, (,), +, -$	Equivalent Java operators. The numerator of the $/$ operator is cast to a `double` to prevent truncation.
`I,M,C,V,EN,` `EC,MC`	These variables are given the associated values of the polymer properties they represent (e.g. `I` will get the value of the number of half-initiators in the molecule, `C` the number of crosslinkers, etc).
`l`	The length of a single segment in the polymer chain.
`w`	The weight of the polymer. Equivalent to `I`$*w_I+$`M`$*w_M+$`C`$*w_C$.

```java
 1 public class EquationImplementation implements Equation {
 2
 3   public double evaluate
 4   (
 5     long I, long M, long C,
 6     long V, long EN, long EC, long MC
 7   ){
 8     return
 9       Math.PI*((double)(4))/3*Math.pow((0.3)*
10       Math.sqrt(((double)((I+M+2*C-V)))/6),3)
11     ;
12   }
13
14 }
```

Figure 6.3: Example of a generated Java function for evaluating a custom expression. Lines 9 and 10 were generated using an ANTLR parser. The rest of the EquationImplementation class serves as a template.

Recognition (ANTLR) grammar describing mathematical equations with implied multiplication, which can be found in Appendix B. ANTLR is a parser generator for reading and processing structured text files [62]. The ANTLR grammar we created was used to generate a parser which has been implemented in such a way that it creates an equivalent Java equation for a given mathematical equation. The Java equation is used, in combination with a class template, to create a compilable Java class. An instance of the class is generated by runtime compiling the class using the JANINO library [63].

The elements that can be used in the mathematical equation, and the Java counterparts are listed in Table 6.3. Note that both `l` and the weights used in `w` are also listed as parameters in the configuration file. By separating these parameters from the equations, they can be changed without having to alter the equations. We deemed this solution more user-friendly, as changing the equation is expected to be more error-prone. Another thing to be noted is the casting of the enumerator in the equation to a `double`. This prevents truncation, which would happen when two non-decimal numbers are divided in Java, e.g. `2/4==0`, but `((double)2)/4==0.5`. Figure 6.3 shows the generated implementation of the `evaluate` function. This is the Java equivalent of the mathematical equation found in Figure 6.2 and Table 6.2. To be able to invoke the generated code we need to define the structure before compiling, since the structure would otherwise only be known after implementation, which is runtime. This has been done by implementing the `Equation` interface, which can be found in Figure 6.4.

```
public interface Equation{

  public double evaluate
  (
    long I, long M, long C,
    long V, long EN, long EC, long MC
  );

}
```

Figure 6.4: Java interface used to describe the structure of the EquationImplementation class, which is shown in Figure 6.4, before a compilable Java definition is available.

6.3 Output

To satisfy Requirement R 3, as defined in Section 1.4, we do not only need to support dynamic input, but dynamic output processing as well. Different properties can be extracted from the simulation state. Among these properties should at least be polydispersity, molecule size and a full molecular weight distribution, so we can compare the results of our simulation with the lab results of [38].

6.3.1 Custom output processing

We use the string equation to Java object conversion described in Section 6.2.3 for custom output processing. Among the set of input parameters, shown in Table 6.1, is a set $\mathcal{F}$ of comma-separated custom output equations. The custom equation structure allows us to calculate different properties of a polymer. This is also the limitation of the custom equation, since we do not want the value of the equation for each polymer in our simulation; this would be a rather large amount of output data to process. Instead, we offer multiple alternatives using modifiers for custom *output* equations, which enclose the custom equation:

- **AVG**: calculate the average value of custom equation $f \in \mathcal{F}$ over all polymers in the simulation state, i.e.
$$\text{AVG}(f) : \{P_1, ..., P_K\} \to \frac{\sum_{i=1}^K c_i f_i}{\sum_{i=1}^K c_i}.$$
- **WAV**: calculate the weighted average value of custom equation $f \in \mathcal{F}$ over all polymers in the simulation state, i.e.
$$\text{WAV}(f) : \{P_1, ..., P_K\} \to \frac{\sum_{i=1}^K c_i f_i w_i}{\sum_{i=1}^K c_i w_i}.$$
- **ZAV**: calculate the z-average value of custom equation $f \in \mathcal{F}$ over all polymers in the simulation state, i.e.
$$\text{ZAV}(f) : \{P_1, ..., P_K\} \to \frac{\sum_{i=1}^K c_i f_i w_i^2}{\sum_{i=1}^K c_i w_i^2}.$$
- **SUM**: calculate the sum of values of custom equation $f \in \mathcal{F}$ over all polymers in the simulation state, i.e.
$$\text{SUM}(f) : \{P_1, ..., P_K\} \to \sum_{i=1}^K c_i f_i.$$

With

$$w_i : S_i \in \{S_1, ..., S_k\} \to \mathbb{R}^*$$

defined as

$$\langle I_i, M_i, C_i, V_i, EN_i, EC_i, MC_i \rangle \mapsto w_I I_i + w_M M_i + w_C C_i.$$

This is the weight function `w` defined in Table 6.3. Each one of these modifiers can be combined with any number of the following ones, in any order:

- **BIN**: calculate not the average or sum over *all* molecules, but rather over each weight class (e.g (1-2] Dalton, (2-3] Dalton, etc.). The size of these weight classes increases exponentially, meaning that data generated with the BIN modifier should be shown on a logarithmic scale. The name of this modifier refers to the process of binning, or grouping data in categories.
- **INC**: include non-polymers (initiator radicals, monomers and crosslinkers) in the data as well.
- **EXC**: exclude biggest molecule in data

In addition to these modifiers we have added the following default functions, which make obtaining commonly used properties easier:

- **MWD**: molecular weight distribution, alias for BIN(SUM(`w`))
- **MN**: number average molar mass, alias for AVG(`w`)
- **MW**: weight average molar mass, alias for WAV(`w`)
- **MZ**: z-average molar mass, alias for ZAV(`w`)
- **PDI**: polydispersity index, or MW/MN, which is not an alias as it cannot be recreated using allowed syntax

6.3.2 Data extraction

Output is given at a fixed conversion interval. Each time this interval is reached, the full set of molecule species is passed to the custom output functions. All output functions extract the necessary data and calculate the values for their equations. This data is saved to be used later in the data conversion step, which is further explained in Section 6.3.3.

Molecule structures When running a simulation with molecule structures, more output is potentially generated at the end of a conversion interval. One of the input parameters is a comma-separated set of maximum 3D model sizes (in number of atoms). Once the size of the structural model of the biggest molecule in the simulation is smaller or equal to one of the maximums, but the structural model of the result of a simulated chemical reaction is larger than this maximum, the (previously) biggest molecule model is converted to a format readable by viewers of 3D molecule models.

Size limits 3D molecule viewers are limited in the size of molecules they can render, which is the reason for the usage of maximums instead of minimums. The widely used MOL format only supports molecules up to 999 atoms due to syntax restrictions [64]. An alternative 3D format called Protein Data Bank (PDB) format supports molecules up to 99,999 atoms syntax-wise, but is not as well supported as MOL [65]. Since both formats have their drawbacks 3D models are converted to both formats when possible.

Structural model to 3D model conversion Converting structural models to the MOL and PDB file formats is made trivial by using the conversion algorithm described in Section 5.4.3. However, two additional steps need to be performed:

1. 3D molecule models restrict node labels to atom codes. For this reason we replace the graph labels by the codes shown in Table 6.1.
2. Atoms in 3D molecule models have coordinates, whereas our graph model does not. For this reason we assign coordinates to the nodes.

Whereas the former is trivial, the latter is not, as the layout of the 3D models is limited in one way: atoms cannot occupy the same exact coordinate. We have chosen to align the 3D models of our molecule structures on a grid to prevent collisions between coordinates. An example of a 3D model with a grid layout is shown in Figure 6.5a. Note that each polymer chain has its own unique x-coordinate. The second half of each crosslinker and the half-crosslinkers with pendent vinyl groups have been given their own z-layer, with the purpose of making it easier to see crosslinks and pendent vinyl groups

The grid layout however, is not a realistic representation of a molecule structure. Rather than implementing an algorithm providing a better layout of our 3D model, we rely on third party tools. The ’3D optimization’ feature of ChemSketch 3D viewer “aims to reliably reproduce reasonable conformations from (possibly very unreasonable) 2D drawings” [66]. Figure 6.5b shows the 3D-optimized version of the model with the grid layout. It is clear that this is a more realistic layout, as connections between the nodes are not stretched anymore. Figure 6.5c shows that the 3D-optimization is also a viable option for larger models.

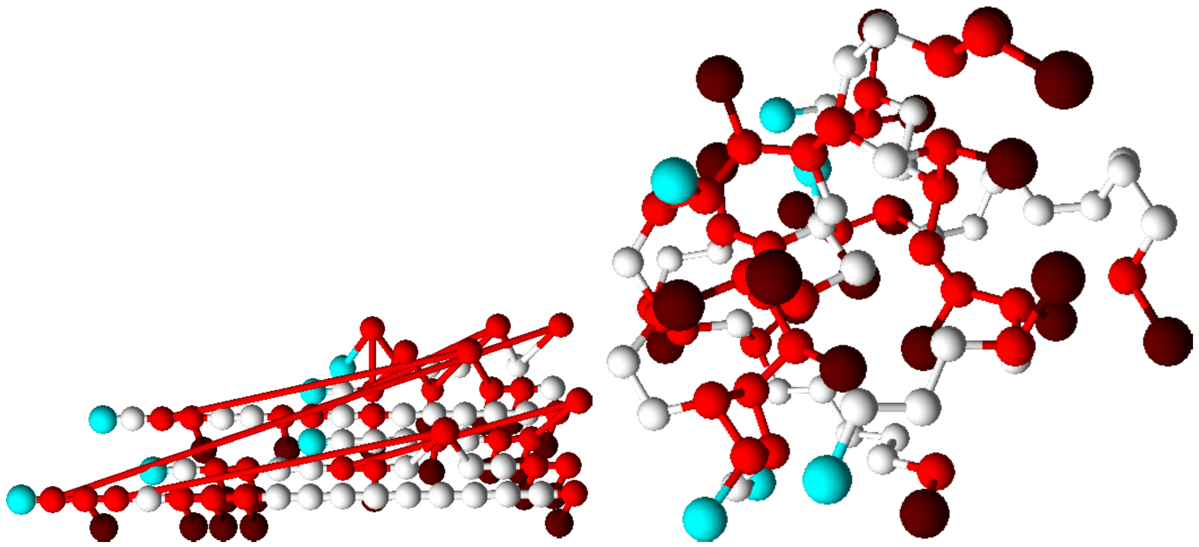
6.3.3 Data conversion

Whereas the 3D models are outputted as soon as the data is extracted, all other data is only outputted at the end of the simulation. The data that is collected by custom output functions during the simulation is divided into two categories: overall data and data per weight category. We will refer to these categories as **non-binned** and **binned** data.

Non-binned data Non-binned output functions only create a single data point per conversion interval, and as such generate a fixed-sized list of data during the simulation. This makes it easy to combine all these data lists are combined into a single output file. We have chosen to use the Comma-Separated Values (CSV) file format, since it is easy to use and widely supported because of its simplistic file structure. The first line of the file contains the percentages of conversion at which data has been extracted from the simulation. Every line after that is filled with the name and data from a custom function.

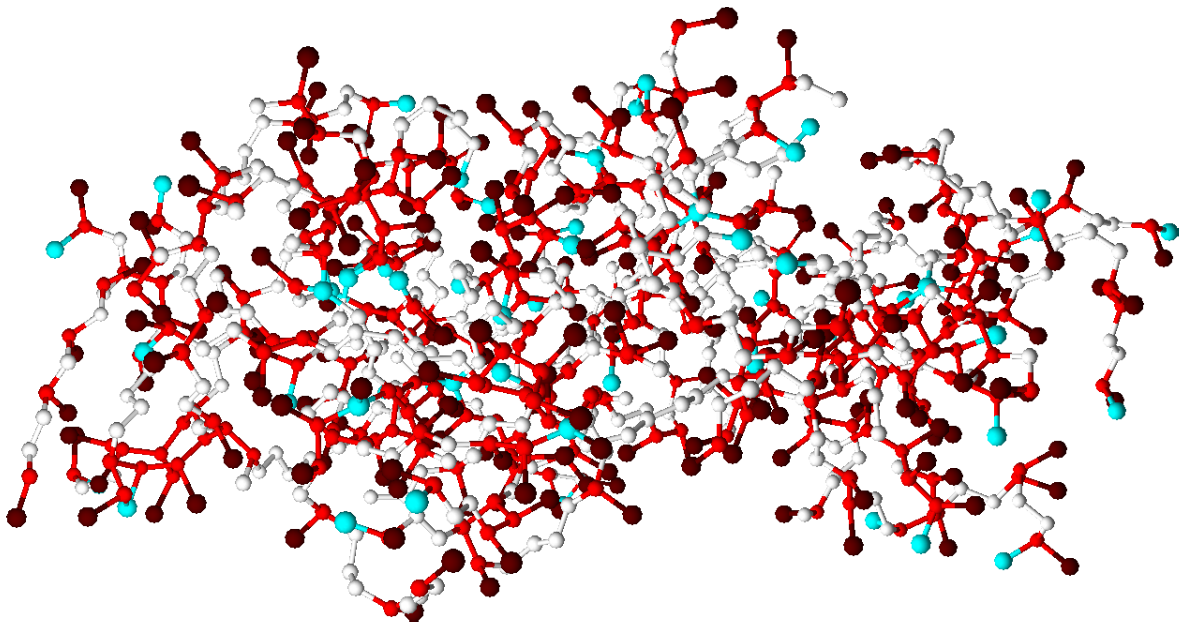
Binned data The binned output contains data for multiple weight categories, giving us a matrix with a data point for each weight category at each point of conversion. This prevents the binned data from being added to the non-binned data. Instead, each set of binned data is converted to a separate CSV file. Similar to the non-binned data, the binned data used the first row for conversion percentages. The first column contains the lower bound of each weight range in Dalton. This format is chosen specifically to support Origin, which is analysis software for chemistry. This format is also the main reason that data is first collected and only converted at the end of the simulation. The CSV format makes it easy to add another row, but hard to add another column, as each line in the file would have to be appended separately. Instead we collect the data of the entire simulation, and create the CSV file for the transposed data.

The dynamic input of our simulation, which we discussed in this chapter, allows us to easily change a simulation parameter via an Excel sheet. The generated nature of this Excel sheet makes it possible to add or remove an input parameter by changing a single Java annotation. Furthermore, the dynamic output processing we discussed allows us to extract a plenitude of properties from our chemical simulation. Due to the chosen formats we can use the output files in third party programs, among which are ChemSketch (for 3D models), Origin (for MWDs) and Excel. Together, the dynamic input and output satisfy Requirements R 2 and R 3 defined in Section 1.4.



(a) Grid layout

(b) Model in Figure 6.5a 3D-optimized by ChemSketch



(c) Larger 3D-optimized model

Figure 6.5: 3D models of molecule structures. Red nodes represent half-crosslinkers, dark red nodes represent half-crosslinkers with a pendent vinyl group, white nodes represent monomers and blue nodes represent half-initiators.

Chapter 7

Results and discussion

In this chapter we discuss the simulation results of multiple simulations. The chemical experiments we simulate are those done by M. Schotman in [38]. Schotman has performed multiple polymerization reactions with the initiator azobisisobutyronitrile(AIBN), the crosslinker ethylene glycol dimethacrylate (EGDMA), the RAFT agent 2-propanoic acid butyl trithiocarbonate (PABTC) and the monomers solketal methacrylate (SMA) and glycidyl methacrylate (GMA). The polymerization reactions were performed at a temperature of 70°C. The reaction scheme of this polymerization is given in Appendix A.

In Section 7.1 we discuss the parameters for each simulation that was executed and how these parameters have been obtained. Subsequently, in Section 7.2, we compare our simulation results with the experimentally obtained data of the simulated experiments. Lastly, in Section 7.4, we show how the time complexity of our simulation scales with the number of simulated molecules.

7.1 Simulation parameters

We simulated a total of 16 experiments from the research of Schotman. For each of these experiments the corresponding values of the configuration parameters, listed in Table 6.1, have been entered into our simulator. Several of these parameters are the same over all simulations. These parameters have been listed in Table 7.1.

7.1.1 Fixed parameters

The parameters needed by the simulation algorithm that are shared between simulations are the initiator radical concentration, steric hindrance factors and the function for the interaction volume of a polymer.

We consider AIBN to be a slow initiator, as it takes between five and ten hours for half of the initiator molecules to split into initiator radicals [15,16]. For this reason we enter an initial initiator radical concentration of zero. The half-initiators containing the initial reactive centers of the polymers are thus mainly introduced via the RAFT agent. Each simulation starts with one billion molecules. The steric hindrance factors ψ, ϕ and ω have been obtained from [28].

We use the function $R_H = 0.0144(w_{SEC})^{0.561}$ for both the calculation of interaction volume and polymer diameter. R_H converts the apparent weight of a nanoparticle w_{SEC} in Dalton (Da), experimentally obtained via the Size Exclusion Chromatography (SEC) technique, to the radius of this nanoparticle R_H (in nm) in a solution [67]. This function is chosen over statistical approaches such as [68,69,70], as it yields more realistic values presumably due to it being based on experimental data.

The interaction volume $vol_i = 2.2\frac{4}{3}\pi(R_H)^3 10^{-24}$ is calculated by first converting the radius to a spherical volume (in nm^3) using $\frac{4}{3}\pi(R_H)^3$ and subsequently multiplying it by 10^{-24} to obtain the function for interaction volume in dm^3. The factor 2.2 is a calibration factor which we will discuss in more detail later in this section. Also note that variable l for the length of a polymer segment has not been used in vol_i, but is often used in the statistical approaches. Each subunit in a polymer chain is considered a segment, i.e. each half-initiator, half-crosslinker and monomer that is part of a polymer chain.

The parameters used specifically for structured simulations are rather straightforward. We use the two-letter codes for the noble gases argon (Ar), Helium (He), Neon (Ne) and Krypton (Kr). By using noble gases as the atoms that represent molecules in our 3D-model, we prevent programs such as Chem3D from adding implied extra hydrogen atoms to the structures, as is the case when using carbon atoms as representatives. Furthermore, we limit the size of the 3D-model to 999 atoms, as this is the maximum size for MOL files.

The parameters needed for simulation output processing consist of the weights of the half-initiator, monomer and crosslinker molecules, which are needed to calculate polymer weights, and a set of output functions $\mathcal{F}$. As explained in Section 6.3.1, these functions are used to select the data that is extracted from the simulation. We extract the following properties from the simulation:

- **Z-average particle size**: Schotman used a technique called Dynamic Light Scattering (DLS) to measure polymer size. This technique calculates particle size using a z-average [71]. To reflect this we also use the z-average to calculate particle size, using the ZAV() function defined in Section 6.3.1 in combination with the radius function R_H mentioned previously, which has been multiplied by two to obtain the function for the diameter: "ZAV(0.0288 wˆ0.561)".

- **Molecular Weight Distribution (MWD)**: The weight distribution is not an average, but contains data for each weight category and thus enables us to predict in more detail how the molecular weights shift over time. It is mainly used to get insight into the gelation process.

- **Polydispersity Index (PDI)**: Schotman also obtained the polydispersity using DLS. DLS calculates PDIs differently, using molecule sizes rather than molecular weights, and will be reffered to as Size Polydispersity Index (SPDI) [72]. As a result, the PDIs obtained from our simulation and the DLS experiments are not directly comparable, but should show similar trends. The usage of mass dispersity is chosen over size dispersity, as the former can be calculated more accurately in our simulation.

- **Branching density**: Branching density, also referred to as crosslinking density, can be defined as the number of branching points over the total number of subunits in polymers [73]. This is one of the properties that Schotman was unable to measure [38]. Due to the nature of our simulation, which tracks both the numbers of pendent vinyl groups and crosslinks in each polymer, we are able to calculate this density. To this purpose we use the functions AVG((2C-V)/(I+M+2C-V)) and AVG(C-V), which calculates the average number of branching points per polymer segment and the average number of crosslinks per polymer respectively. The number of segments in a polymer is I+M+2C-V, i.e. the number of half-initiators, monomers and half-crosslinkers minus V, the number of half-crosslinkers that are not yet part of a polymer chain. Note that each half-crosslinker that is part of a polymer chain is counted as a branching point, i.e. two branching points per crosslinks and one per pendent vinyl group.

Table 7.1: Values of used simulation parameters

Parameter	Value	Source
Needed by simulation algorithm		
[I]	0 $g\,l^{-1}$	
c_T	1,000,000,000	
ψ	0.53 for SMA * 0.56 for GMA *	[28,75]
ϕ	0.001455	[28]
ω	0	[28]
vol_i	”2.2*4/3 π (0.0144 wˆ0.561)ˆ3*10ˆ-24”	[67]
l	0.2487	[75]
Needed for simulation with molecule structures		
I_{atom}	Ar	
M_{atom}	He	
C_{atom}	Ne	
P_{atom}	Kr	
$\mathcal{X}$	999	
d	high detail	
Needed for output		
$\mathcal{F}$	{”PDI”,”ZAV(0.0288 wˆ0.561)”,”MWD”, ”AVG((C-V)/(I+M+2C-V))”,”AVG(C-V)”}	
w_I	82.10 Da	[76]
w_M	200.23 Da for SMA 142.15 Da for GMA	[77] [78]
w_C	198.22 Da	[79]

* obtained using linear interpolation

Note that the data does not include the half-initiator, monomer and crosslinker species, as the INC() modifier was not used. This is done to better match the results of the DLS experiments, since the DLS technique was unable to pick up molecules smaller than 8nm in the research of Schotman, like EGDMA, GMA, SMA or AIBN. This will be further discussed in Section 7.2.

7.1.2 Dynamic parameters

The parameters for the concentration of RAFT agent ([RAFT]), monomer ([M]) and crosslinker ([C]) varies between the different experiments. We calculated these concentrations for each experiment performed in [38], using the densities of the solvents, monomers, crosslinker and initiator at 70°C, as they are given in Table 7.2. We were unable to obtain the density of the RAFT agent PABTC at 70°C and used the density at 25°C instead, which is 1264 $g\,l^{-1}$ [74]. For simplicity we assume that the reaction mixture has the same volume as the sum of the volumes of the molecules that make up the mixture. Monomer concentrations were given in weight percentage (wt%), rather than in mole per liter ($mol\,l^{-1}$). This monomer concentration is the combined concentration of monofunctional and difunctional monomers. The calculated concentrations are shown in Table 7.3. Note that Experiments 2 and 10 are typical polymerization reactions for SMA and GMA respectively, and every other one of the experiments have one altered parameter.

7.1.3 Calibration

In Section 2.3.2 we mentioned that gelation occurs when many of the polymers interconnect to form a single network. In the simulation results a similar event can be seen; many polymers merge to form

Table 7.2: Densities at 70°C

Molecule	**Density** ($\mathrm{g\,l^{-1}}$)	**Source**
Solvents		
chlorobenzene	1052*	[80]
dioxane	976.726	[81]
ethyl acetate	838.733	[82]
dimethylformamide	902.25	[83]
Monomers		
solketal methacrylate (SMA)	848*	[84,85]
glycidyl methacrylate (GMA)	1073*	[78,86]
ethylene glycol dimethacrylate (EGDMA)	1015*	[79,87]
Other		
azobisisobutyronitrile (AIBN)	880*	[88,89]

* obtained using linear extrapolation

a large polymer network. However, the gel points initially predicted by our simulation happened at far higher conversion percentages than the actual gel points of the experiments that were simulated. We hypothesize that this is caused by an overestimation of the local concentration of vinyl groups, causing more intramolecular reactions (and thus fewer intermolecular reactions) to occur and thus postpone the gel point. Even though we include each vinyl group in a polymer in the calculation of its local concentration of vinyl groups, not each of these groups has the same probability to react with a reactive center within this polymer. Not only should the vinyl group not be blocked by other parts of the polymer, but the polymer itself must be able to fold in such a way that both the reactive center and the vinyl group meet, in order to be able for them to form a crosslink. The former has already been corrected for by the inclusion of steric hindrance in our chemical model, but the latter has not been taken into consideration in the chemical model.

Remember that the local concentration of vinyl groups in a molecule of species S_i is defined as $[L_i] = v_i/vol_i$ in Equation 4.2. As vol_i is one of the changeable input parameters, it can be used used to correct the overestimation of $[L_i]$ by multiplying it with a correction factor. This correction factor was obtained by performing a one-parameter fit on the gel point. We have defined the gel point as the point of conversion at which the size of the biggest particle in the simulation shows a sharp increase. The calibration itself was performed by simulating Experiment 2 in Table 7.3, and altering vol_i so that the predicted gel point happens at 50% conversion, like in the experiment by Schotman [38]. The resulting correction factor was 2.2 when using the formula for hydrodynamic radius R_H by Pomposo *et al.*, as indicated earlier in Section 7.1.1. For comparison: the correction factors for the radius functions obtained using statistical approaches as given by [68,69,70] ranged from approximately 150 to 250, due to these functions yielding far smaller radii, and thus higher local concentrations. Note that Experiment 2 was performed using the SMA monomer. Calibration should be performed per polymerization reaction. However, no conversions for GMA gel points were available to us, so the correction factor of 2.2 was also used in the simulation of GMA polymerization.

7.2 Simulation results

To analyze the accuracy of our simulator, multiple SMA and GMA polymerizations were simulated. In this section we compare the results of these simulations to the lab results of these polymerization, as given in [38].

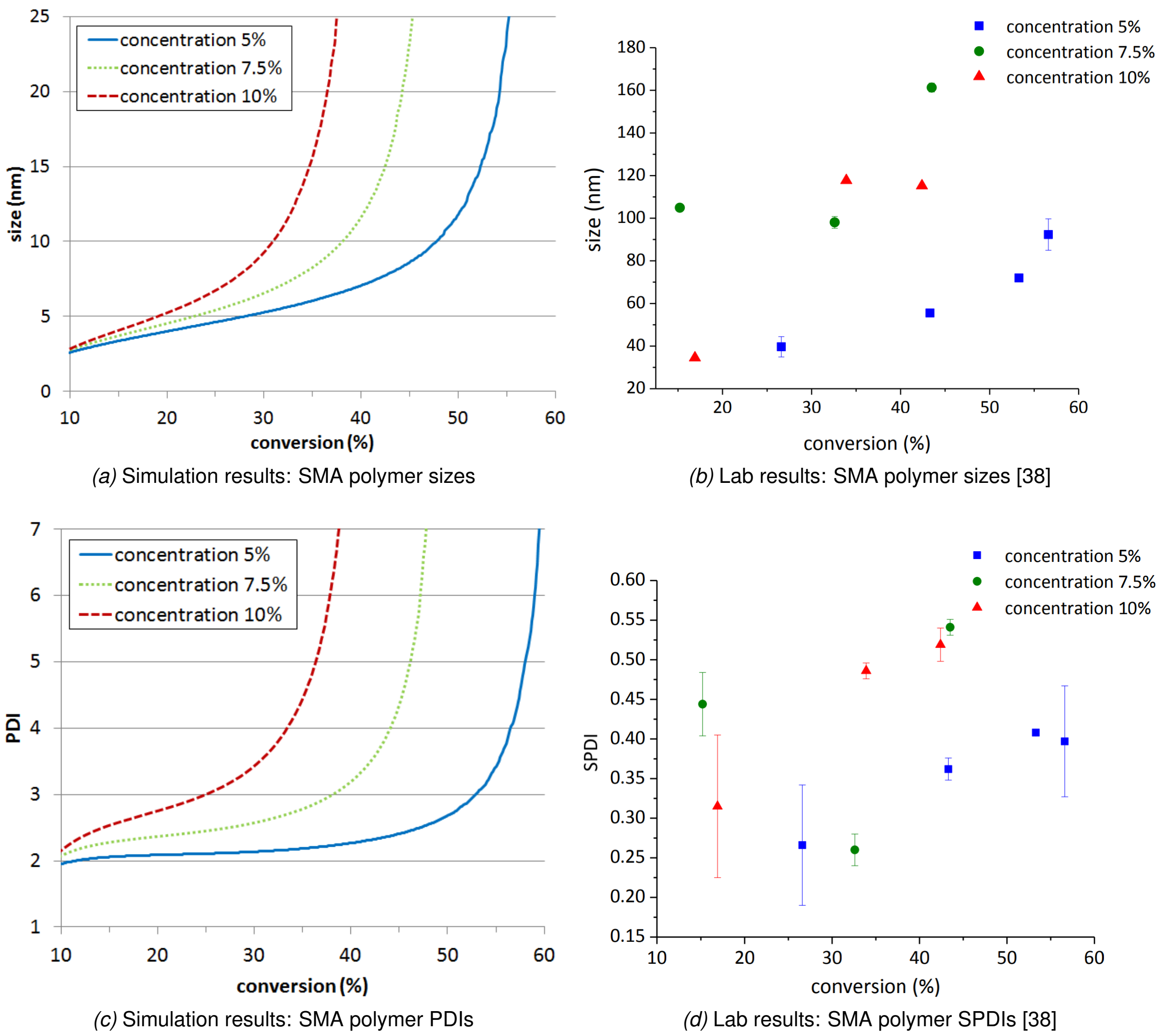
Table 7.3: Concentrations

Experiment	[C]:[M]	[C]+[M]:[RAFT]	monomer concentration (wt%)	solvent	monomer	[Initiator] (mol l^{-1})	[M] (mol l^{-1})	[C] (mol l^{-1})	[RAFT] (mol l^{-1})
1	1:3	50:1	5	chlorobenzene	SMA	8.5×10^{-4}	1.9×10^{-1}	6.5×10^{-2}	5.1×10^{-3}
2	1:3	50:1	7.5	chlorobenzene	SMA	1.3×10^{-3}	2.9×10^{-1}	9.7×10^{-2}	7.6×10^{-3}
3	1:3	50:1	10	chlorobenzene	SMA	2.3×10^{-3}	4.4×10^{-1}	1.5×10^{-1}	1.2×10^{-2}
4	1:3	50:1	7.5	dioxane	SMA	1.2×10^{-3}	2.7×10^{-1}	9.1×10^{-2}	7.1×10^{-3}
5	1:3	50:1	7.5	ethyl acetate	SMA	1.0×10^{-3}	2.4×10^{-1}	7.9×10^{-2}	6.2×10^{-3}
6	1:3	50:1	7.5	dimethylformamide	SMA	1.1×10^{-3}	2.5×10^{-1}	8.5×10^{-2}	6.6×10^{-3}
7	1:3	50:1	5	chlorobenzene	GMA	1.1×10^{-3}	2.5×10^{-1}	8.4×10^{-2}	6.6×10^{-3}
8	1:3	50:1	7.5	chlorobenzene	GMA	1.7×10^{-3}	3.8×10^{-1}	1.3×10^{-1}	9.9×10^{-3}
9	1:3	50:1	10	chlorobenzene	GMA	2.2×10^{-3}	5.0×10^{-1}	1.7×10^{-1}	1.3×10^{-2}
10	1:3	50:1	6	chlorobenzene	GMA	1.3×10^{-3}	3.0×10^{-1}	1.0×10^{-1}	7.9×10^{-3}
11	1:3	25:1	6	chlorobenzene	GMA	2.6×10^{-3}	3.0×10^{-1}	1.0×10^{-1}	1.6×10^{-2}
12	1:3	75:1	6	chlorobenzene	GMA	8.9×10^{-4}	3.0×10^{-1}	1.0×10^{-1}	5.3×10^{-3}
13	1:2	50:1	6	chlorobenzene	GMA	1.3×10^{-3}	2.6×10^{-1}	1.3×10^{-1}	7.7×10^{-3}
14	1:4	50:1	6	chlorobenzene	GMA	1.3×10^{-3}	3.3×10^{-1}	8.2×10^{-2}	8.1×10^{-3}
15	1:3	50:1	6	dioxane	GMA	1.2×10^{-3}	2.8×10^{-1}	9.4×10^{-2}	7.4×10^{-3}
16	1:3	50:1	6	dimethylformamide	GMA	1.1×10^{-3}	2.6×10^{-1}	8.7×10^{-2}	6.9×10^{-3}

7.2.1 Solketal methacrylate (SMA)

We have performed multiple simulations of a polymerization with the SMA monomer. The goal of these simulations was to show the effect of monomer concentration and solvent on the particle size and the polydispersity.

Monomer concentration The polymerization reactions of the SMA monomer and EGDMA crosslinker were simulated at multiple monomer weight concentrations. Simulations were performed where SMA and EGDMA had a combined total of 5, 7.5 and 10$\mathrm{wt}\%$. In Figure 7.1a and 7.1c the predicted polymer sizes and polydispersity are plotted against conversion. Lower monomer concentrations leads to smaller particles. This is the expected result, as a decrease in concentration leads to a decrease in intermolecular reactions, and thus more intramolecular reactions in proportion. These intramolecular reactions lower the number of available pendent vinyl groups in the polymers, and thus lead to fewer possible intermolecular crosslinking reactions, which in turn leads to smaller particles. This is corroborated by experimental data as seen in Figure 7.1b, where a monomer concentration of 5$\mathrm{wt}\%$ clearly results in the smallest particles.



(a) Simulation results: SMA polymer sizes

(b) Lab results: SMA polymer sizes [38]

(c) Simulation results: SMA polymer PDIs

(d) Lab results: SMA polymer SPDIs [38]

Figure 7.1: Comparison between sizes and PDIs yielded by different monomer concentrations (in percentage of the total mixture weight) using SMA. These are the results of Experiments 1,2 and 3 in Table 7.3.

The lab results for the higher monomer concentrations are not conclusive however, due to experimental inaccuracies. As an effect no conclusion can be drawn on whether 7.5 or 10$\text{wt}\%$ monomer concentration yields bigger particles. The predicted PDI follows the increase in size, meaning that the weight differences become larger as conversion and polymer size increase, which is again corroborated by lab data.

For these experiments we also predicted the gel points by plotting conversion against the size of the largest polymer, as seen in Figure 7.2. The point at which the plot line becomes nearly vertical is what we consider the gel point. The predicted gel points for 10, 7.5 and 5$\text{wt}\%$ are measured at approximately 42, 50 and 61$\%$ conversion respectively. This closely resembles the gel points measured by Schotman, which had a conversion of 43, 50 and 60$\%$ respectively. Note that the gel point of the polymerization with 7.5$\text{wt}\%$ monomer weight, with a conversion of 50$\%$, was the gel point that was used to calibrate the simulation, as mentioned in Section 7.1.3.

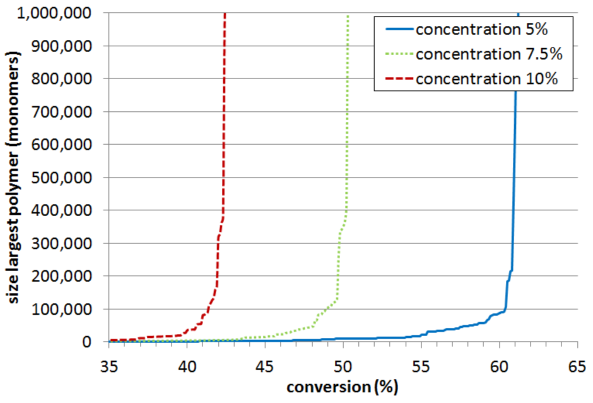


Figure 7.2: Size of largest polymer (in number of monomers) in the simulation for multiple monomer weight concentrations. These are the results of Experiments 1,2 and 3 in Table 7.3.

These gel points are also visible in the MWDs. Figure 7.3 plots the polymer weight against the total weight of polymers of that weight for multiple points of conversion during the simulation of Experiment 2. This allows us to see the predicted shift in polymer weights over the course of the polymerization reaction. Polymer weights are again given in Dalton. Peak values shift toward larger polymer weights as conversion increases and the number of smaller molecules decreases. Past 50$\%$ conversion we see a decrease in the number of heavy polymers and the emergence of a second peak. This peak represents a single polymer, as the polymer weight and the total weight of polymers of that weight are approximately equal, with a total weight of 1.5×10^{10} Da in the weight range 1.3×10^{10} to 1.6×10^{10} Da. Not shown in Figure 7.3 is the second peak at 70$\%$ conversion with a weight of 3.8×10^{10} Da.

To get more insight into the emergence of this second peak we also plotted the MWDs at several points of conversion between 50 and 60$\%$ conversion, as can be seen in Figure 7.4. Multiple smaller peaks can be seen at 50$\%$ conversion for polymers of a weight between 3×10^7 and 4×10^7 and between 6×10^7 and 7×10^7 Da, both with a total weight of approximately 7×10^7. We can thus conclude that there are 3 heavy polymers at this point in the simulation. At 51$\%$ conversion there is still a smaller peak between 2×10^7 and 3×10^7Da with a total weight of 5×10^7Da. There are thus again 3 heavy polymers, but the difference in weight between them has increased, as the second peak is of a polymer with a weight of 5.6×10^8Da.

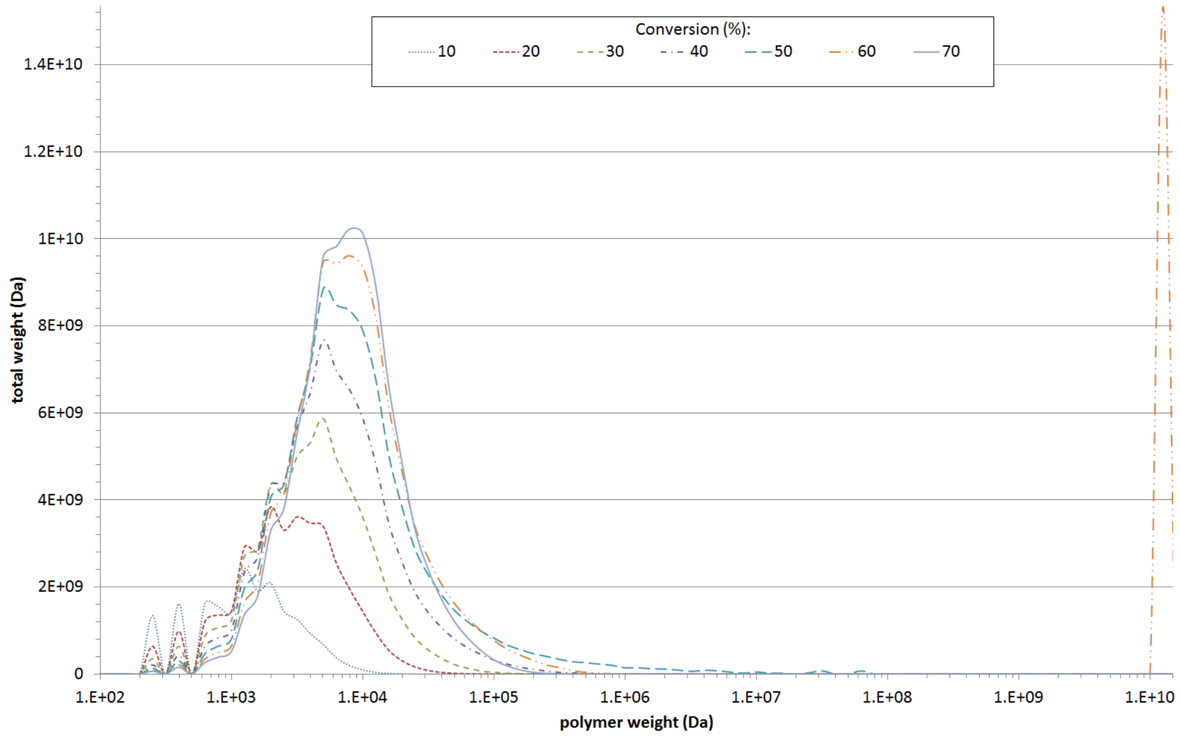


Figure 7.3: MWDs for multiple points of conversion. These are the results of Experiment 2 in Table 7.3.

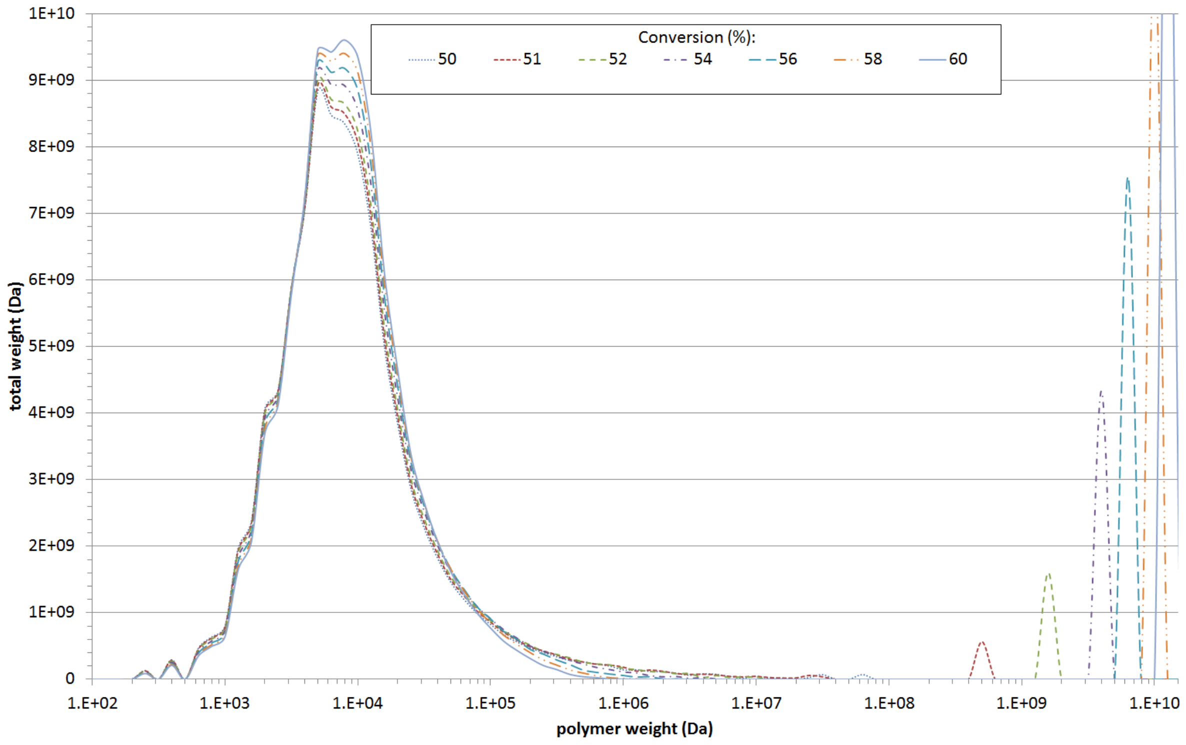
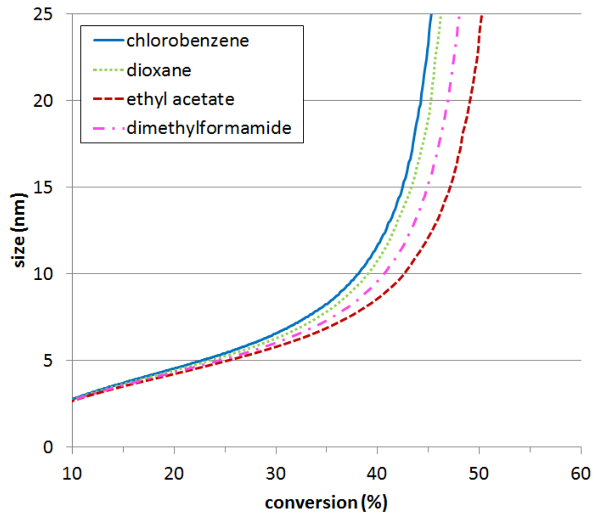


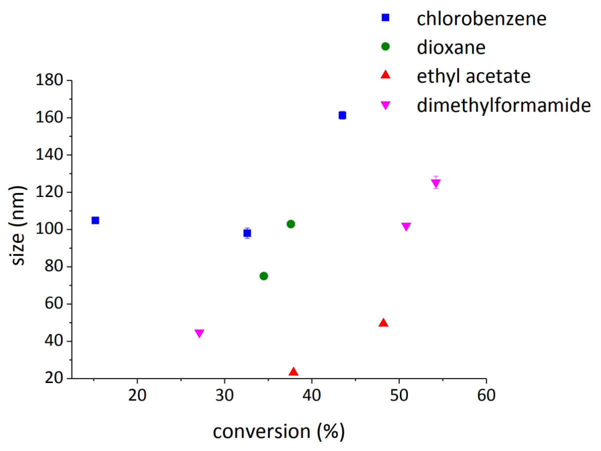
Figure 7.4: MWDs for multiple points of conversion around gel point. These are the results of Experiment 2 in Table 7.3.

At 52% conversion the smaller peaks are gone altogether and the heaviest polymer has nearly tripled in weight. The heavier polymers all aggregate into the heaviest polymer as conversion further increases; the heaviest polymer increases in weight while the number of polymers heavier than 5×10^{10}Da decreases. This is expected, as heavier polymers have a higher probability to react due to these polymers containing more reactive groups.

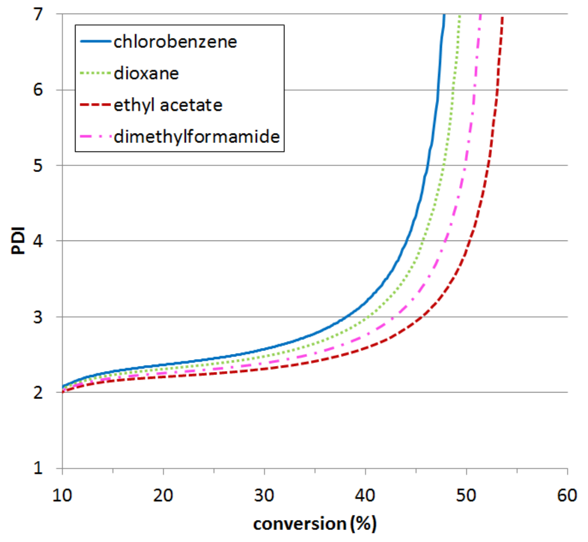
Solvents The polymerization of SMA nanogels in different solvents has been simulated. We mentioned previously in this section that solvent also has an effect on the polymer sizes, due to the absorption of solvent by the polymers. Such effects are not included in our simulation, as the only influence the solvent has is the changing of the total mixture volume and thus the molar concentrations of the half-initiator, RAFT agent, monomer and crosslinker molecules. This is caused by differences in solvent densities, as shown in Table 7.2. Surprisingly, the differences in predicted sizes and PDIs roughly match those found in the lab data, as can be seen in Figure 7.5.



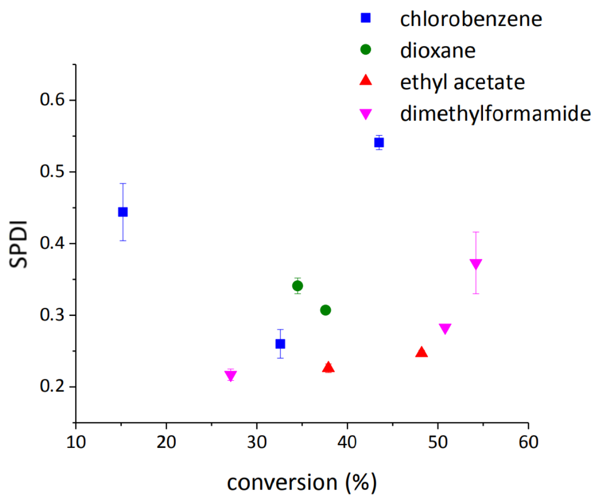
(a) Simulation results: SMA polymer sizes



(b) Lab results: SMA polymer sizes [38]



(c) Simulation results: SMA polymer PDIs



(d) Lab results: SMA polymer SPDIs [38]

Figure 7.5: Comparison between sizes and PDIs yielded by different solvents using SMA. These are the results of Experiments 2,4,5 and 6 in Table 7.3.

Ethyl acetate yields the smallest particles, followed by dimethylformamide, dioxane and chlorobenzene respectively. The PDIs are again lower for smaller polymer sizes, with ethyl acetate having the lowest PDI, followed by dimethylformamide, dioxane and chlorobenzene. This seems to indicate that the measured differences between the solvents are at least partly caused by the way the molar concentrations of the molecules are chosen. It might be worthwhile to perform additional experiments with different solvents where the molar concentrations remain constant, rather than the monomer weight concentration. The simulation input would be equivalent for different solvents in that case, and the results of these simulations would thus be the same for each solvent.

In summary, we were able to successfully predict gel points and trends in both polymer size and PDI for different monomer concentrations in an SMA polymerization. Further successful predictions of trends were made for different molar concentrations caused by differences in solvent densities. Predicted MWDs shows us what happens during gelation in our simulation, showing the emergence of the gel particle and simultaneous reduction of the number of larger polymer. This indicates that the simulator is suitable for gel point prediction and giving further insight into the polymerization process.

7.2.2 Glycidyl methacrylate (GMA)

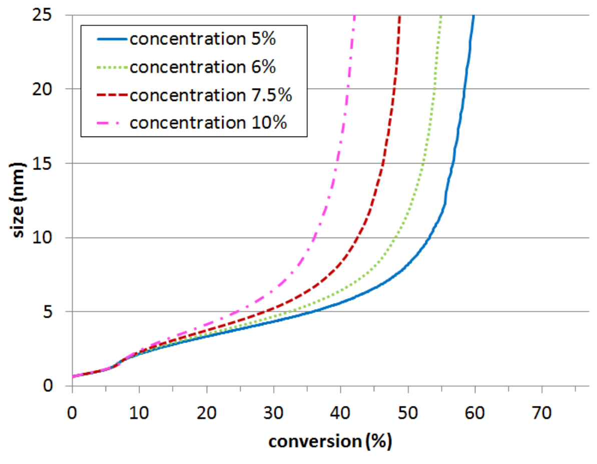
Additional simulations were performed of polymerization with the GMA monomer. The goal of these simulations was to again show the effect of monomer concentration and solvent on the particle size and the polydispersity. Further simulations were performed to also show the effects of the ratio of monomer to crosslinker and the ratio of monomer and crosslinker to RAFT agent.

Monomer concentration In Figure 7.6 the polymer sizes and PDIs are shown for different monomer weight concentrations. Simulations were performed where GMA and EGDMA had a combined total of 5, 6, 7.5 and 10$\text{wt}\%$ of the total mixture weight. Similar to the simulations of SMA polymerizations, we predict an larger polymer sizes and higher PDIs for higher concentrations. Like Schotman, we also conclude that at a concentration of 5$\text{wt}\%$, polymer sizes at the same conversion are smaller than those at concentrations of 7.5 and 10$\text{wt}\%$.

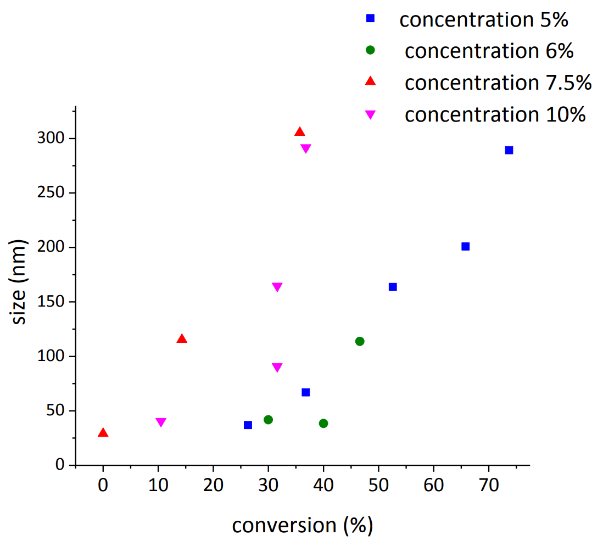
There are however, two major differences with the lab results:

- Predicted polymer sizes of GMA were smaller than those of SMA, whereas the reverse is seen in the lab data. This could be caused by the absence of aspects like polymer solubility in our chemical model which can have a significant impact on polymer sizes [90,91].

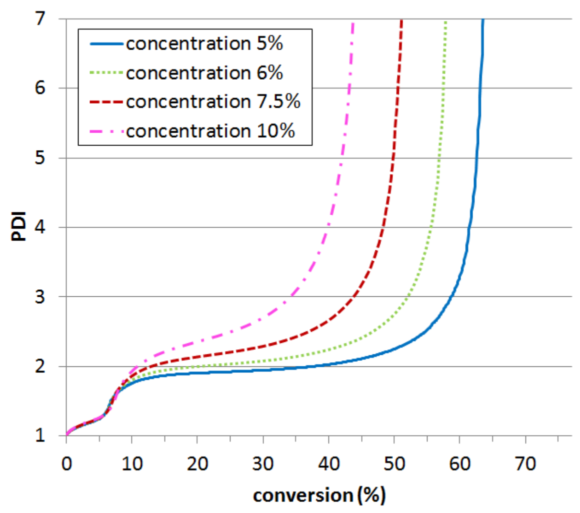
- At a concentration of 5$\text{wt}\%$ no significant increase in SPDI is measured, whereas our simulation does predict an increase in PDI. This is caused by our prediction of a gel point around 65$\%$ conversion. The forming of the gel greatly increases the differences in weight between polymers, and higher PDIs are thus yielded at the gel point. No gelation occurred in the lab experiments however. This could be a sign that the calibration of our simulation using SMA gel points might be insufficient for simulations of GMA polymerization, as previously mentioned in Section 7.1.3.



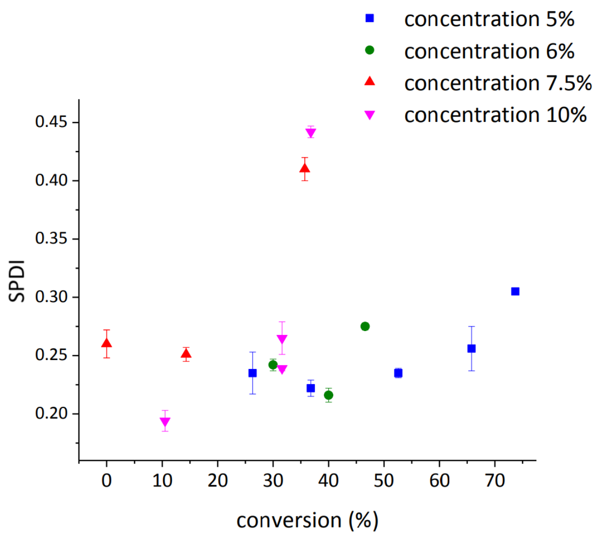
(a) Simulation results: GMA polymer sizes



(b) Lab results: GMA polymer sizes [38]



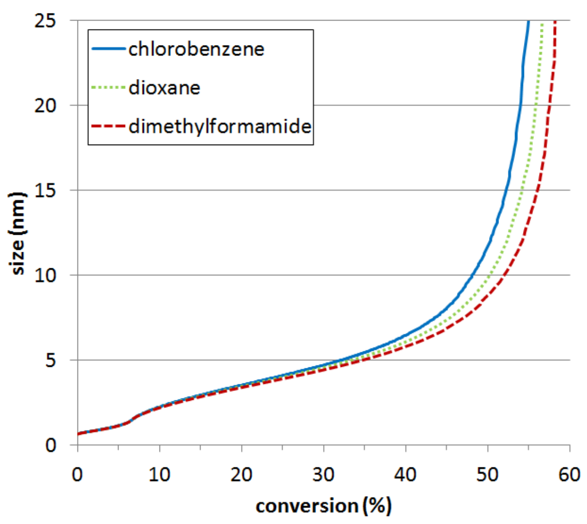
(c) Simulation results: GMA polymer PDIs



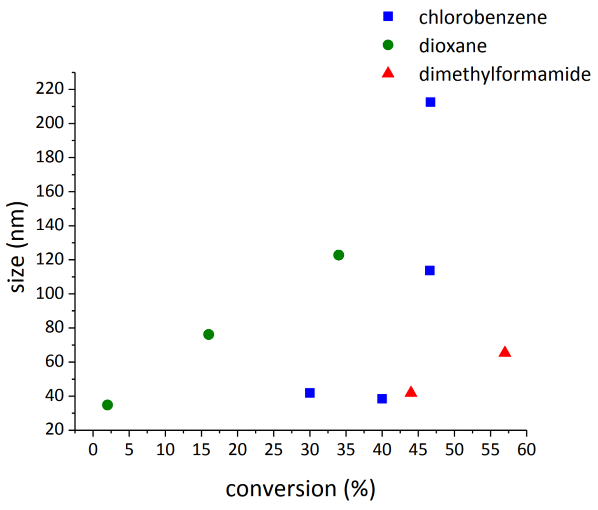
(d) Lab results: GMA polymer SPDIs [38]

Figure 7.6: Comparison between sizes and PDIs yielded by different monomer concentrations using GMA. These are the results of Experiments 7,8,9 and 10 in Table 7.3.

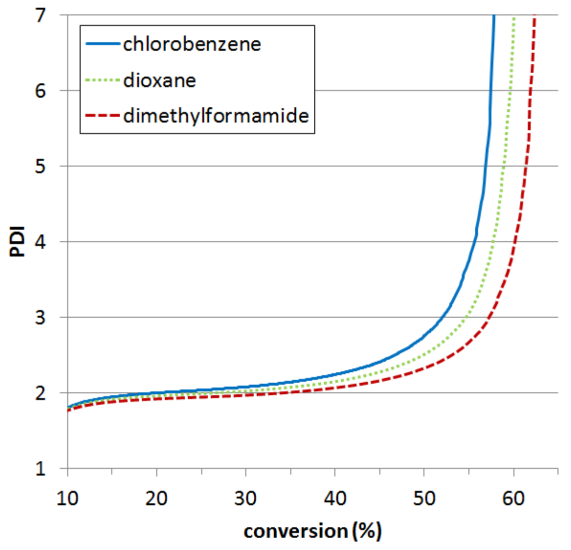
Solvents In Figure 7.7 the polymer sizes and PDIs are shown for different solvents. The simulation results predict the largest polymers and the highest PDI for chlorobenzene, followed by dioxane and dimethylformamide. This is not in agreement with the lab results, as those indicate that dioxane would yield the largest polymers and highest SPDI. It thus seems that the influence of solvent density on the molar concentrations of the initial molecules does not play as big a role in GMA polymerization as in SMA polymerization. The differences cannot be explained just by swelling either, as Schotman has compared the differences in size for a GMA polymer in different solvents, yielding the biggest polymer sizes in chlorobenzene $(102 \pm 0.625\text{nm})$, followed by dimethylformamide $(87.3 \pm 0.905\text{nm})$ and dioxane $(75.5 \pm 0.775\text{nm})$. It thus seems that our simulation was unable to provide further insight into the lab results, which were inconclusive about why polymerization in dioxane yielded bigger polymers at the same point of conversion.



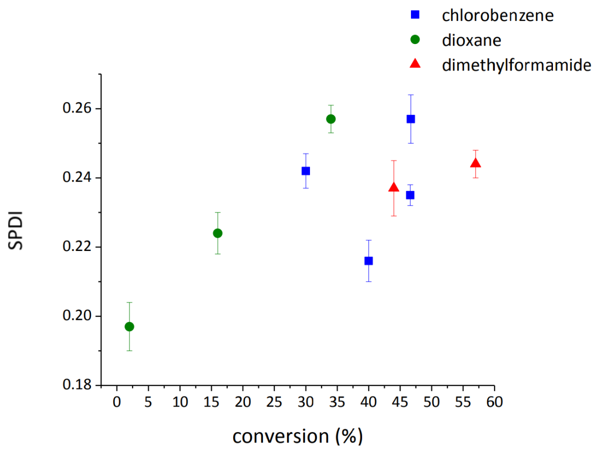
(a) Simulation results: GMA polymer sizes



(b) Lab results: GMA polymer sizes [38]



(c) Simulation results: GMA polymer PDIs

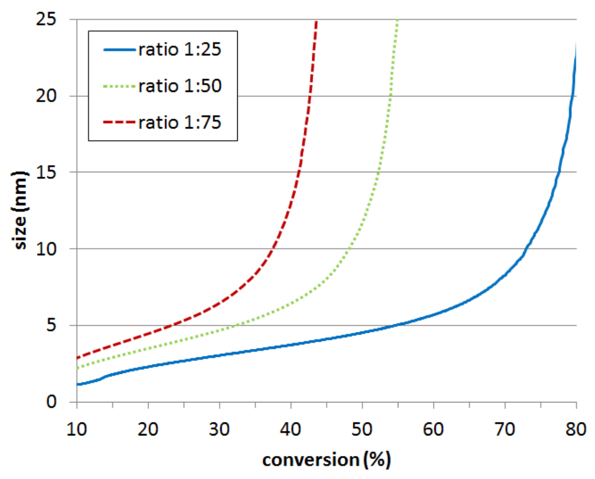


(d) Lab results: GMA polymer SPDIs [38]

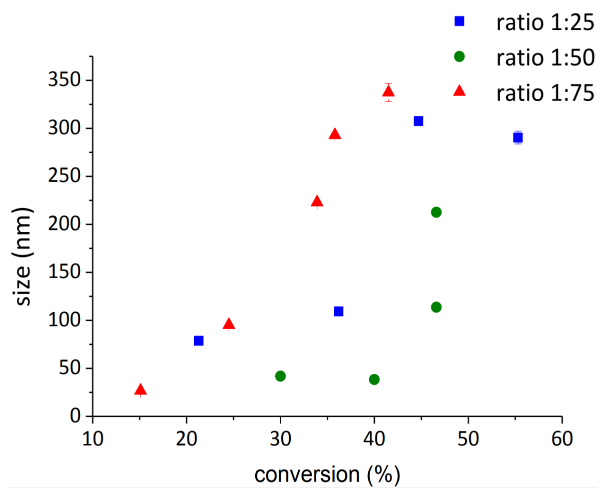
Figure 7.7: Comparison between sizes and PDIs yielded by different solvents using GMA. These are the results of Experiments 10,15 and 16 in Table 7.3.

Ratio RAFT agent to monomer and crosslinker The ratio of the RAFT agent to the monomer and crosslinker was varied to show the effect of the RAFT agent concentration on polymer size and PDI. Ratios of 1:25, 1:50 and 1:75 were simulated. The difference between the simulations of these reactions is mainly the number of initial half-crosslinkers added via the RAFT agent, leading to more reactive centers. The simulation results, as seen in Figure 7.8, predict the lowest polymer size for the same conversion at the highest ratio of 1:25. This is expected, as roughly the same number of monomers has reacted to a higher number of reactive centers, leading to fewer monomers in each polymer chain. Larger polymer size are predicted at the ratio of 1:50 and the largest polymers at the lowest ratio of 1:75. The trends in PDI again roughly follow the trends in polymer size, with the ratio of 1:25 yielding the lowest PDI, followed by the ratios 1:50 and 1:75 respectively.

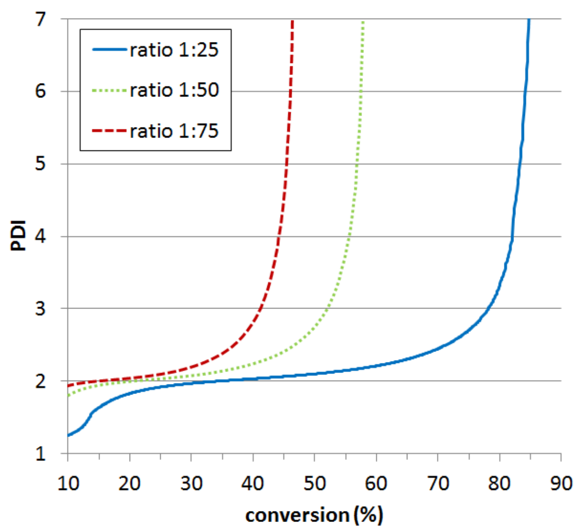
Lab results also show the biggest polymers and highest SPDI at a ratio of 1:75 in comparison to a ratio of 1:50. Lower SPDI and polymer sizes were expected for a ratio of 1:25, like predicted by our simulation, but were not observed, as Schotman remarked. Additional data points are needed to conclude whether a ratio of 1:25 or 1:50 yields the smallest polymers and lowest PDI.



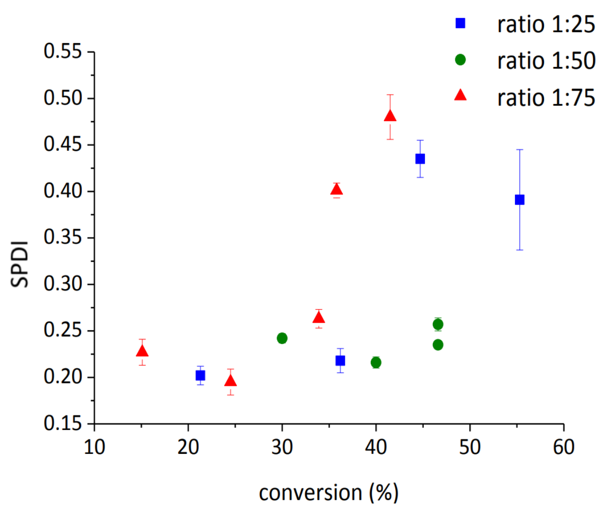
(a) Simulation results: GMA polymer sizes



(b) Lab results: GMA polymer sizes [38]



(c) Simulation results: GMA polymer PDIs

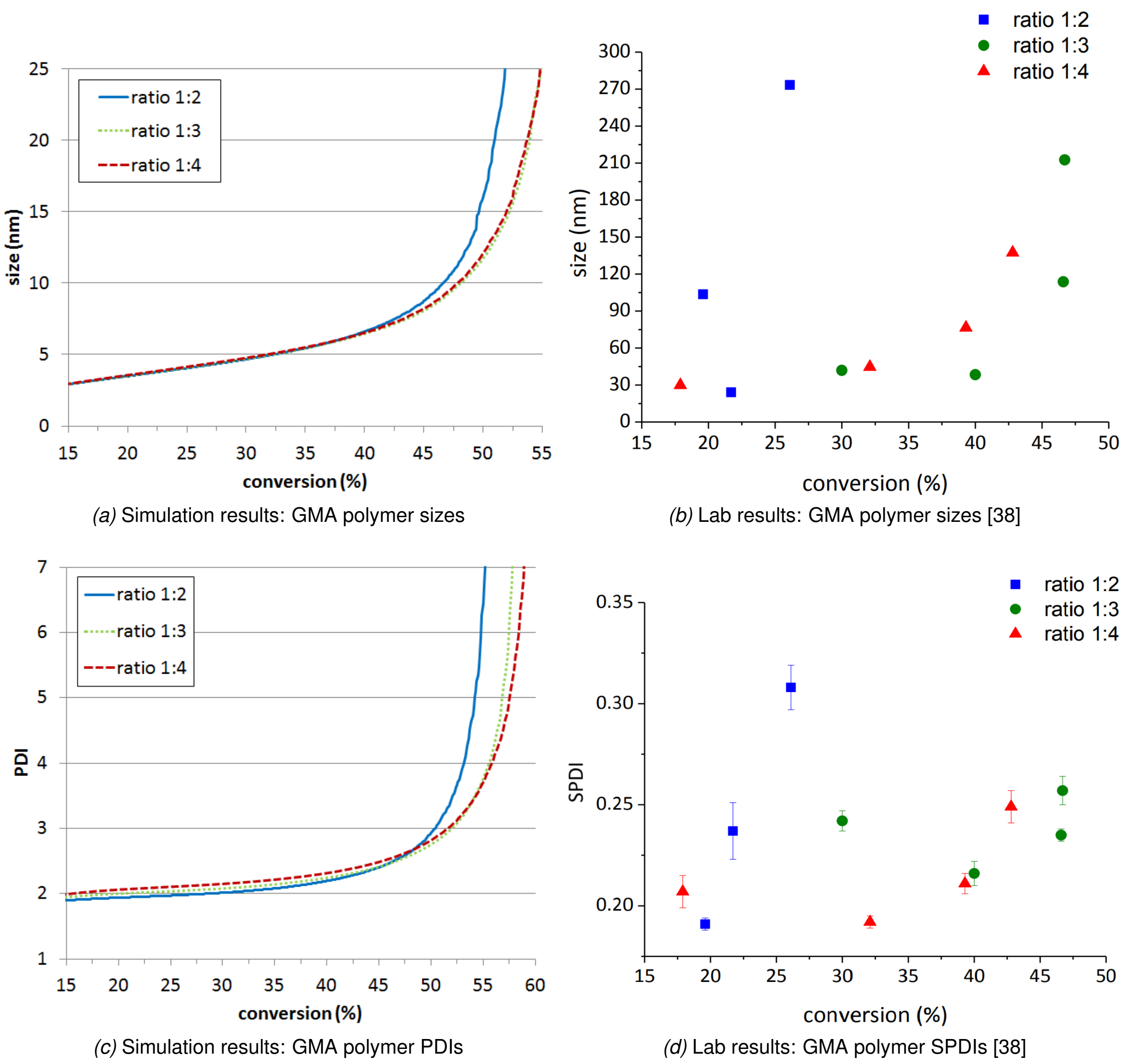


(d) Lab results: GMA polymer SPDIs [38]

Figure 7.8: Comparison between sizes and PDIs yielded by different ratios between the (molar) concentration of RAFT agent and monomers using GMA. These are the results of Experiments 10,11 and 12 in Table 7.3.

Ratio crosslinker to monomer The ratio of crosslinker to monomer plays an important role in the structure of nanoparticles. The higher the concentration of crosslinker in comparison to the monomer, the higher the amount of branching that occurs. At higher ratios of crosslinker to monomer we would thus expect more branches per polymer. Furthermore, we would also expect a higher number of pendent vinyl groups, leading to more intermolecular crosslinking and thus larger particles. The ratio between crosslinker and monomer varied between 1:2, 1:3 and 1:4. The sizes and PDIs against conversion of both the simulation and lab results are shown in Figure 7.9. Against expectations, both the lab and simulation results shows similar sizes and PDIs for the ratios 1:3 and 1:4. Both also show the lowest polymer sizes and PDI for a ratio of 1:2.

However, the polymer size and SPDI for the ratio 1:2 increases at a lower conversion in the lab results. A similar result is only achieved after disabling intramolecular crosslinking in our simulation, yielding a gel point around 30% conversion. A possible explanation could be that the high branching density of 1:2 leads to more rigid particles that do not have the flexibility for intramolecular crosslinking. When intramolecular reactions are not possible, more pendent vinyl groups remain in the polymers, which can then undergo intermolecular reactions, leading to higher sizes at lower conversion percentages.

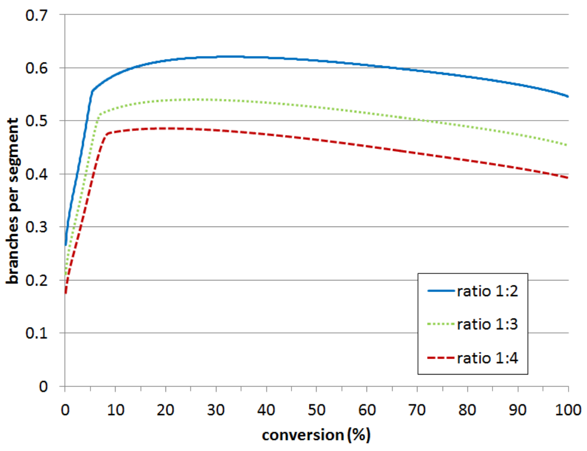


(a) Simulation results: GMA polymer sizes
(b) Lab results: GMA polymer sizes [38]
(c) Simulation results: GMA polymer PDIs
(d) Lab results: GMA polymer SPDIs [38]

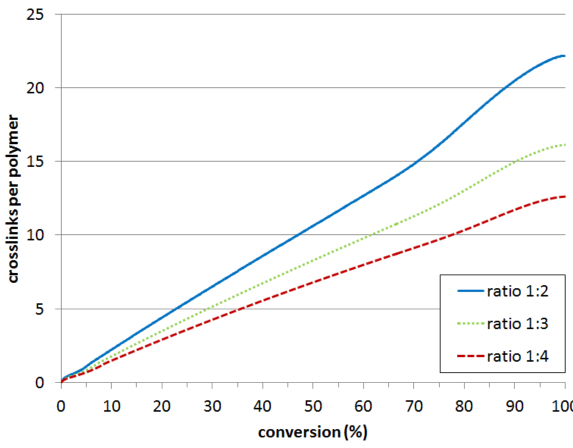
Figure 7.9: Comparison between sizes and PDIs yielded by different ratios between the (molar) concentrations of monomer and crosslinker using GMA. These are the results of Experiments 10,13 and 14 in Table 7.3.

Further simulation results were obtained to examine the amount of branching per polymer and the number of crosslinks per polymer. These results can be found in Figure 7.10. The simulation results indicate that even though the polymer sizes are similar at crosslinker to monomer ratios of 1:3 and 1:4, both the number of crosslinks and the amount of branching is higher for the ratio 1:3. For visual feedback 3D models were generated for each ratio as well, which are shown in Figure 7.11.

The 3D models show an open structure at a ratio of 1:4, a structure with a denser core, but a more open outer shell at a ratio of 1:3 and an altogether denser structure at a ratio of 1:2.

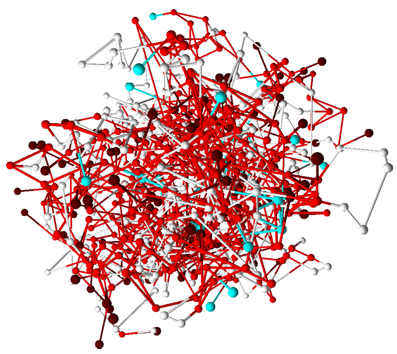


(a) Average branching density

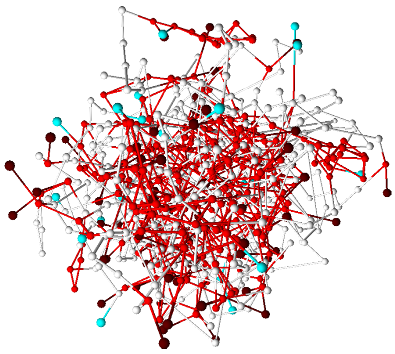


(b) Average number of crosslinks per polymer

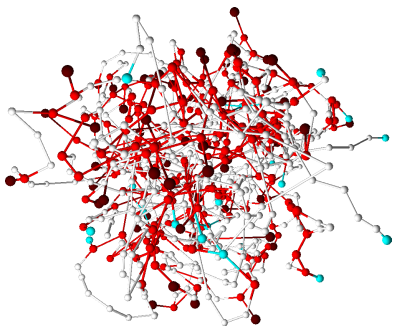
Figure 7.10: Comparison between number of crosslinks and branching densities of different ratios between the concentrations of monomer and crosslinker using GMA. These are simulation results of Experiments 10,13 and 14 in Table 7.3. Figure 7.10a shows the branching density in number of branching points per polymer segment. Each monomer, half-initiator and each half of a crosslinker that is part of a chain is counted as a segment in the polymer.



(a) Ratio 1:2



(b) Ratio 1:3



(c) Ratio 1:4

Figure 7.11: 3D models of polymers for different ratios between the concentrations of monomer and crosslinker using GMA. These are simulation results of Experiments 10,13 and 14 in Table 7.3. Like before, red nodes represent half-crosslinkers, dark red nodes represent half-crosslinkers with a pendent vinyl group, white nodes represent monomers and blue nodes represent half-initiators.

In summary, simulation results of a GMA polymerization showed expected trends in polymer size and PDI. Larger polymer sizes were predicted for higher monomer concentrations, lower RAFT ratios and higher ratios of crosslinker to monomer. Furthermore, our simulation was able to give further insight into crosslink density for different crosslinker-monomer ratios, a property that is hard to measure in the lab, but trivial to obtain using the dynamic output of our simulation. This renders the simulation promising for supporting nanogel research.

7.3 Size differences

There seems to be one main difference between the simulation results and the lab results: sizes predicted by our simulation are significantly lower than those measured in lab experiments. This could be caused by multiple factors:

- Molecules smaller than 8 nanometers were not detected by the DLS measurements of Schotman. This could be caused by limitations of the DLS technique. For particles smaller than 100nm the intensity of the scattered light depends on the particle size. Due to the higher scattered intensity, the signals of larger particles can masks that of smaller particles and the sizes obtained by DLS strongly reflect the sizes of larger particles as a result [92,93].

  Even though we do not include the initial molecules (i.e. half-initiators, monomers and crosslinkers) in our size calculations, we *do* include polymers that are smaller than 8 nanometers. This leads to a lower calculated z-average size.

- A crosslinked polymer is capable of increasing its volume severalfold by absorbing large amounts of solvent [13]. Schotman has shown that the size of a GMA polymer can differ by a factor of two between different solvents, showing sizes ranging from 49nm in chloroform to 111nm in acetonitrile. This swelling caused by solvent absorption has not been taken into account in the calculation of polymer sizes.

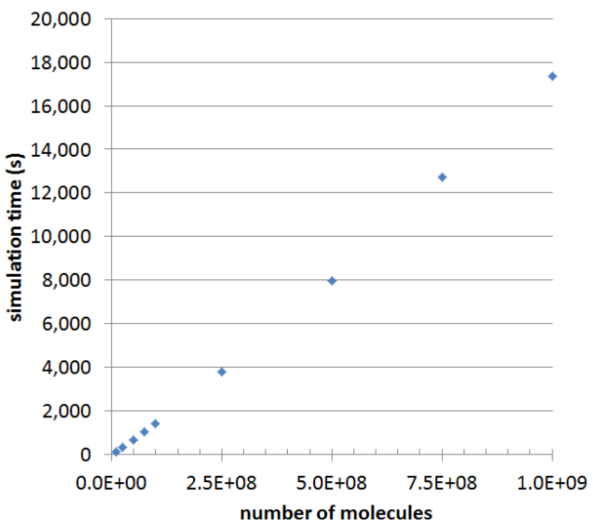
- We mentioned previously that the size function $R_H = 0.0144(w_{SEC})^{0.561}$ converts polymer weights (in Da), as obtained from SEC experiments, to polymer radii (in nm) [67]. However, the weights obtained via SEC also include the solvents that are absorbed by the polymer particles, as the polymers are dissolved during SEC measurements [94]. The weight of these solvents is not included in the calculation of polymer weights by our simulation. Lower polymer weights lead to lower polymer sizes when using R_H, and the resulting sizes are thus an underestimation of the actual sizes.

Several approaches for including these factors are suggested in Section 8.3. Due to the differences in simulated and experimental sizes we recommend comparing size predictions by our simulation with other size predictions by our simulations, rather than with those obtained via lab data.

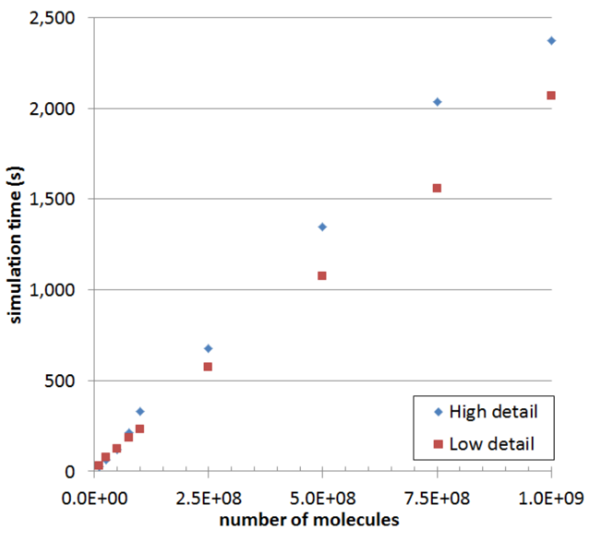
7.4 Scaling

A great advantage of our simulation compared to state-of-the-art systems is its scalability. To quantify this scalability multiple simulations were performed. Experiment 2 in Table 7.3 has been simulated both with a structured and unstructured simulation for multiple numbers of starting molecules to show how the simulator scales in time. Additional simulations were performed using a billion starting molecules to show changes in simulation speed and memory usage over the course of a simulation. The simulations were performed on a computer with an Intel i7-3770 CPU with a clock speed of 3.4 GHz and 8GB of RAM. The Java heap size was limited to 5.5GB.

Simulation time Figure 7.12 shows how the simulation time scales almost linearly with the number of initial molecules c_T in the simulation. This is the case for both the structured and unstructured simulation. Furthermore note that the use of the low detail level for structured simulations decreases the simulation time. The differences between the simulation times of the low and high detail level are not that large however, indicating that the extra abstraction introduced by the the usage of the low detail level is not necessary for successful simulation.



(a) Total simulation time for different numbers of initial molecules c_T.



(b) Total simulation time of structured simulation for different numbers of initial molecules c_T.

Figure 7.12: Simulation times of both the structured and unstructured simulation. Figure 7.12b shows the differences in simulation time for the two detail levels defined in Section 5.2.4.

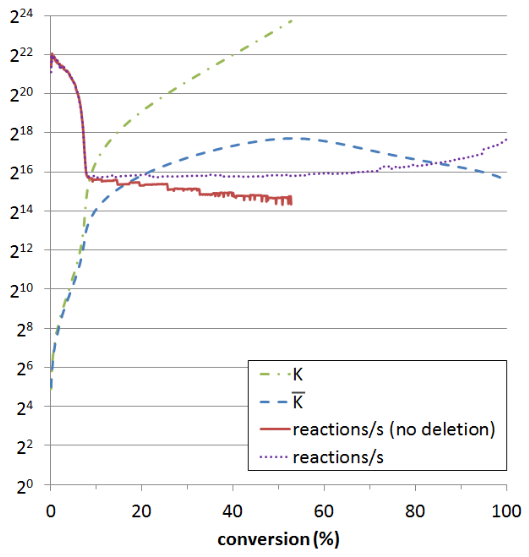
Simulation speed We also compared the simulation speed, in number of simulated reactions per second, of an unstructured simulation, a special unstructured simulation where we disabled deletion of empty molecule species, and structured simulations on both high and low detail.

Figure 7.13 shows how the simulation speed sharply decreases at the start of the simulation. This is caused by the increase in reactive centers in the mid-chain position, causing more reactions to be rejected, thus reducing the number of simulated reactions per second. This figure also shows that the number of non-empty molecule species peaks around the gel point. The simulation speed increases again after the gel point, as the number of molecule species decreases.

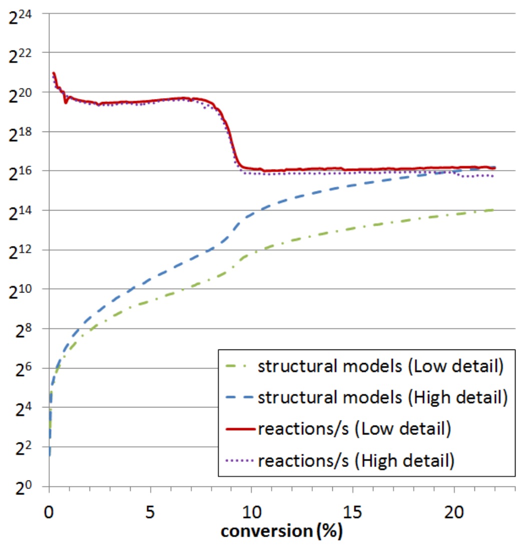
Further analysis of reaction rejection was performed. A total of 97.8% of the selected reactions was rejected, accounting for a total of roughly 86% of the simulation time. Making the simulation rejection-free could thus lead to a speedup of roughly 600%.

As expected the simulator used far more memory when not deleting empty molecule species and ran out of memory at around 52% conversion. However, the results of this incomplete simulation clearly show the relation between the number of tracked molecule species and simulation speed. Dips in speed are clearly visible when the BITs are doubled in size, as this doubling increases the time needed for sampling and updating the BITs. The outliers in simulation speed at the end of the simulation are presumably caused by the Java garbage collector, as more memory needs to be freed as the memory limit is approached.

Figure 7.13b shows the influence of detail level on the speed of a structured simulation. It is clear that the use of a lower detail level has no significant impact on the simulation speed, even though the number of unique structural models is greatly lowered. Furthermore note that the simulation ended at 23.5% conversion as that is when the maximum 3D model limit $\mathcal{X}$ of 999 was reached. The structured simulation shows similar performance as the unstructured simulation until this point of conversion.



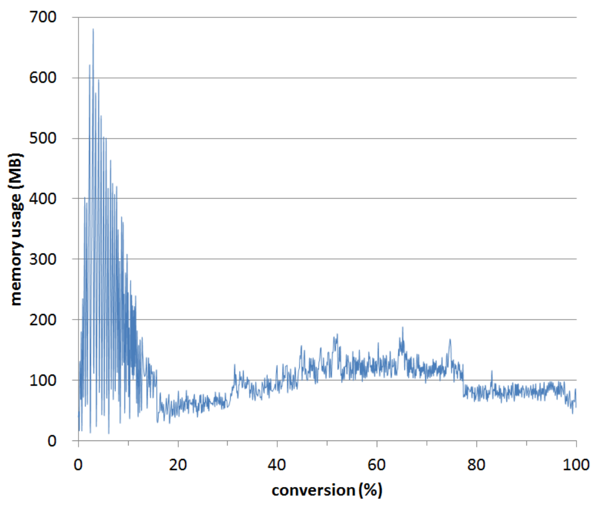
(a) Simulation speed of unstructured simulation.



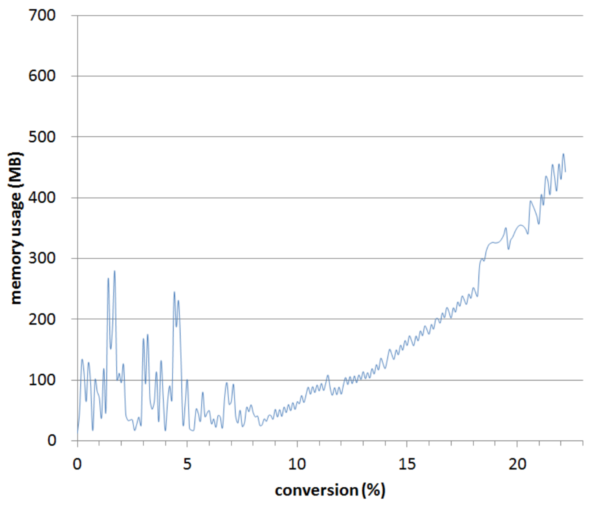
(b) Simulation speed of structured simulation.

Figure 7.13: Simulation speeds of both structured and unstructured simulation with a billion initial molecules c_T. Figure 7.13a shows how the simulation speed depends on $\overline{K}$, the total number of different molecule species in the simulation. The reaction speed in these graphs is given in simulated reactions per second. For comparison the graph in this figure also shows the simulation speed of a simulation where non-empty species were not deleted, and the number of total molecule species K. Figure 7.13b shows the differences in simulation speed for the two detail levels defined in Section 5.2.4 and the total number of structural models when using these detail levels.

Simulation memory usage The memory usage of both a unstructured and structured simulation is plotted against conversion in the graphs in Figure 7.14. The memory usage peaks at the start of the simulation. This is presumably partly caused by the higher simulation speed, meaning that more unreachable Java objects are created before the Java garbage collector frees up memory by deleting these. As simulation speed decreases, so does the memory usage. This is clearly visible in the memory usage of the structured simulation which lowers when the simulation speed decreases at around 8.5% conversion. As expected, the structured simulation consumes more memory and the memory usage seems to increase faster as the size of the structural models increases.



(a) Memory usage of unstructured simulation.



(b) Memory usage of structured simulation.

Figure 7.14: Memory usage of both the structured and unstructured simulation with a billion initial molecules c_T. This is the memory used as obtained from the Java Runtime Environment and does not include reserved memory by the Java Virtual Machine.

Chapter 8

Conclusions and recommendations

8.1 Contributions and requirement fulfillment

The simulator described in this work fulfills the requirements listed in Section 1.4. We used a chemical model that combines our own model for intramolecular crosslinking, as discussed in Section 4.3.3, with the steric hindrance model by Tripathi *et al.*, as presented in [28], thus fulfilling Requirement R 1.

We also created a method to read and generate Excel files from program code, allowing for both easy simulation configuration by the user as well as effortless changes to the number of simulation parameters of our simulation. This was discussed in Section 6.2. As one of our simulation parameters was a function that converts a molecule to the interaction volume of that molecule, a parser for such mathematical equations was created. As the user can thus change all necessary simulation parameters, we consider Requirement R 2 fulfilled.

This parser was further extended for the extraction of data from the simulation, so it allows the user to describe which properties to obtain from a set of molecules. This allows for the calculation of mass polydispersity, molecular weight distributions, average polymer size and crosslink density, as we have shown in Sections 6.3 and 7.1.1. We also created a time and memory efficient structural model, which can be converted to 3D molecule models, as described in Section 5.4. This covers all necessary output types to fulfill Requirement R 3. As mentioned in Section 6.3.2, data points are calculated at fixed conversion intervals, thus giving multiple data points over the course of a simulation and fulfilling Requirement R 5.

The last requirement, R 4, has also been fulfilled, as shown in Section 7.4. Efficient scaling was achieved by our novel approach of tracking molecular reactivities rather than reaction rates, as described in Section 5.2.2. These reactivities were tracked using Binary Indexed Trees (BITs). The use of binary trees is not uncommon in KMC simulations [95]. However, the use of BITs, which have lower memory usage than regular binary trees, is a novel approach.

8.2 Conclusions

We have successfully created a simulator for crosslinked RAFT copolymerization that fulfills the requirements listed in Section 1.4.

The simulation algorithm used by the simulator was created specifically for the simulation of crosslinked polymerization and scales almost linearly in time with the initial number of molecules in the simulation. This speed was obtained by:

- tracking the reactivity of molecule species rather than the rates of reactions between species

- using BITs to track these reactivities, which can be both sampled and updated in logarithmic time
- only tracking non-empty molecule species

This is all possible under the assumption that the reactivity of reactive groups is independent of the molecule in which they are located. Our approach could thus also be used for the efficient simulation of other branching polymerization reactions, as this assumption applies to polymerization reactions in general.

Due to the dynamic output of the simulator we were able to obtain not only typical results like molecular weight distributions, polydispersity indices, gel points and molecule sizes, but also properties that are harder to obtain in lab experiments, like branching density and average number of crosslinks per polymer. These hard to obtain results together with the visual feedback from 3D structures generated by our simulation gave more insight into the polymerization process. One remaining issue is the conversion of polymer weights to polymer sizes, as the function that was used for this yielded sizes far lower than those measured in the lab. This function for polymer size is entered in the simulation configuration, and can be altered without changing the implementation of the simulator. When comparing the predicted polymer sizes with other predicted polymer sizes, similar trends were observed to those in the lab results. These results are thus qualitative rather than quantitative.

8.3 Recommendations

We recommend additions to the simulation algorithm to improve the following aspects of the simulation.

- **Simulation speed**, by implementing the rejection-free version of our algorithm, as presented in Appendix C. Roughly 97% of selected reactions are not simulated, accounting for roughly 86% of the simulation time. The simulation speed can thus be greatly improved by only selecting reactions that are simulated.
- **Simulation detail**, by implementing the extension of our algorithm, as presented in Sections C.2 through C.4 in Appendix C. This makes the calculation of other aspects of the simulation possible, like:
  - reaction time, allowing us to predict conversion over time
  - RAFT agent addition and fragmentation reactions, allowing us to better simulate different RAFT agents
  - initiator splitting, allowing us to not only simulate initiators that split near instantly and initiators that split slowly, but also those in between

  The addition of reaction time will also allow for easier comparisons between lab results and simulation results, since the calculation of the conversion takes effort for lab samples, but tracking the reaction time is trivial.
- **Simulation accuracy**, by adding a filter to remove molecules that are not detectable in the lab. In Section 7.2 we mentioned that molecules smaller than 8 nanometers were not visible in DLS results, even though these molecules were present in the lab samples. This is one of the possible factors causing a difference between the sizes predicted by our simulations and sizes measured in the lab. Adding a filter excluding these smaller molecules from our simulation results could thus improve the accuracy of our simulation.

Further improvements to simulation accuracy can be made by improving vol_i, the simulation parameter for the function of the interaction volume of a polymer:

- For the creation of our interaction volume function we used the size function created by Pomposo *et al.* as presented in [67], which predicts polymer sizes and uses the weights of polymers, as obtained via Size Exclusion Chromatography (SEC), as input. Weights obtained via SEC include the weight of the solvent absorbed by the polymers. Including this additional weight when calculating polymer weights should thus lead to more accurate predictions of polymer sizes.

  A function could be created for the calculation of solvent absorption of polymers, which could be combined with the function for polymer sizes. This function should have multiple parameters which influence solvent absorption, like solvent density, solvent and monomer polarity, branching density and polymer weight. The latter two of these can be calculated by our simulator, as we have shown previously. Effectively, the combined function would be able to predict polymer swelling for different solvents.

- Size functions, as obtained by statistical approaches such as those given by [68,69,70], multiply the size of a linear polymer containing the same number of polymer segments with a factor for the size decrease caused by an increase in density due to branching. This factor is calculated using the number of crosslinks in the particle, giving smaller sizes for higher rates of branching. As the size function used in this work calculated polymer sizes using only polymer weights, this size function could be improved by also using the rate of branching as a parameter in a similar manner.

An interesting follow-up study would be to examine biodegradation of nanoparticles by randomly removing monomers in the 3D-models yielded by our simulation, mimicking the (triggered) degradation of monomers in the body. This could give further insight into the sizes of the particles that are yielded after degradation, which is an important factor for the removal of nanoparticles from the body after it has served its purpose. It would also give insight into whether degradation would lead to a more open structure at different monomer to crosslinker ratios. A more open structure would be important for drug release, as drugs particles can easier leave the nanocarrier when the pores in the carrier become larger.

Another interesting follow-up would be the simulation of other polymerization reactions. Not only is our simulation algorithm efficient for the simulation of any polymerization reaction, it can also be easily extended due to the nature of the Monte Carlo simulation. By changing the set of initial molecules to include a set of starting polymers, the simulator could be used to simulate other types of polymerizations, like star-polymerizations, copolymerizations with crosslinkers with more than two vinyl groups or even complex nanoparticles like silicon quantom dots, which can contain over fifty vinyl groups. This would require only minor changes to the code if a similar chemical model can be used.

Bibliography

[1] P. J. Mohr, D. B. Newell, and B. N. Taylor, "CODATA Recommended Values of the Fundamental Physical Constants: 2014," *Journal of Physical and Chemical Reference Data*, jul 2015. [Online]. Available: http://arxiv.org/abs/1507.07956http://dx.doi.org/10.1103/RevModPhys.88.035009

[2] J. Poly, D. Wilson, M. Destarac, and D. Taton, "A comprehensive investigation into controlled/living chain growth crosslinking copolymerization including a back to basics modeling," *Journal of Polymer Science Part A: Polymer Chemistry*, vol. 47, no. 20, pp. 5313–5327, oct 2009. [Online]. Available: http://doi.wiley.com/10.1002/pola.23580

[3] J. Wang, J. D. Byrne, M. E. Napier, and J. M. DeSimone, "More Effective Nanomedicines through Particle Design," *Small*, vol. 7, no. 14, pp. 1919–1931, jul 2011. [Online]. Available: http://doi.wiley.com/10.1002/smll.201100442

[4] A. Albanese, P. S. Tang, and W. C. Chan, "The Effect of Nanoparticle Size, Shape, and Surface Chemistry on Biological Systems," *Annual Review of Biomedical Engineering*, vol. 14, no. 1, pp. 1–16, 2012. [Online]. Available: http://www.annualreviews.org/doi/10.1146/annurev-bioeng-071811-150124

[5] J. Wang, W. Mao, L. L. Lock, J. Tang, M. Sui, W. Sun, H. Cui, D. Xu, and Y. Shen, "The Role of Micelle Size in Tumor Accumulation, Penetration, and Treatment," *ACS Nano*, vol. 9, no. 7, pp. 7195–7206, jul 2015. [Online]. Available: http://pubs.acs.org/doi/10.1021/acsnano.5b02017

[6] H. Cabral, Y. Matsumoto, K. Mizuno, Q. Chen, M. Murakami, M. Kimura, Y. Terada, M. R. Kano, K. Miyazono, M. Uesaka, N. Nishiyama, and K. Kataoka, "Accumulation of sub-100 nm polymeric micelles in poorly permeable tumours depends on size," *Nature Nanotechnology*, vol. 6, no. 12, pp. 815–823, 2011. [Online]. Available: http://dx.doi.org/10.1038/nnano.2011.166

[7] H. Cabral, J. Makino, Y. Matsumoto, P. Mi, H. Wu, and T. Nomoto, "Systemic Targeting of Lymph Node Metastasis through the Blood Vascular System by Using Size - Controlled Nanocarriers," no. 5, pp. 4957–4967, 2015.

[8] H. Gao, W. Shi, and L. B. Freund, "Mechanics of receptor-mediated endocytosis," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 27, pp. 9469–9474, 2005. [Online]. Available: http://www.pnas.org/content/102/27/9469.abstract

[9] S. V. Vinogradov, T. K. Bronich, and A. V. Kabanov, "Nanosized cationic hydrogels for drug delivery: Preparation, properties and interactions with cells," *Advanced Drug Delivery Reviews*, vol. 54, no. 1, pp. 135–147, 2002.

[10] F. Kiessling, M. E. Mertens, J. Grimm, and T. Lammers, "Nanoparticles for Imaging: Top or Flop?" *Radiology*, vol. 273, no. 1, pp. 10–28, 2014. [Online]. Available: http://pubs.rsna.org/doi/10.1148/radiol.14131520

[11] J. Grimm and D. A. Scheinberg, “Will nanotechnology influence targeted cancer therapy?” *North*, vol. 29, no. 10, pp. 1883–1889, 2008.

[12] F. Alexis, E. Pridgen, and L. K. Molnar, “Factors Affecting the Clearance and Biodistribution of,” *Molecular pharmaceutics*, vol. 5, no. 4, pp. 505–515, 2008.

[13] S. Nandi and H. H. Winter, “Swelling behavior of partially cross-linked polymers: A ternary system,” *Macromolecules*, vol. 38, no. 10, pp. 4447–4455, 2005.

[14] J. Ramos, M. Pelaez-Fernandez, J. Forcada, and A. Moncho-Jorda, “CHAPTER 4. Nanogels for Drug Delivery: the Key Role of Nanogel & Drug Interactions,” in *Soft Nanoparticles for Biomedical Applications*, 2014, pp. 133–156. [Online]. Available: http://ebook.rsc.org/?DOI=10.1039/9781782625216-00133

[15] Sigma-Aldrich, “Applications : Free Radical Initiators,” pp. 3–4. [Online]. Available: http://www.pnas.org/content/108/23/E196.short

[16] L. Wako Pure Chemical Industries, “Azo Polymerization Initiators Comprehensive Catalog,” p. 33, 2016. [Online]. Available: https://www.wako-chemicals.de/files/download/pdf/wako_azo_polymerization_initiators_catalog_25.pdf

[17] G. Moad, E. Rizzardo, and S. H. Thang, “Living radical polymerization by the RAFT process,” *Australian Journal of Chemistry*, vol. 58, no. 6, pp. 379–410, 2005.

[18] M. Semsarilar and S. Perrier, “’Green’ reversible addition-fragmentation chain-transfer (RAFT) polymerization,” *Nature Chemistry*, vol. 2, no. 10, pp. 811–820, 2010. [Online]. Available: http://www.nature.com/doifinder/10.1038/nchem.853

[19] “Polymer Science RAFT : Choosing the Right Agent to Achieve Controlled Polymerization,” pp. 2–5, 2016. [Online]. Available: https://www.sigmaaldrich.com/technical-documents/articles/materials-science/polymer-science/raft-polymerization.html

[20] S. Perrier and P. Takolpuckdee, “Macromolecular design via reversible addition-fragmentation chain transfer (RAFT)/xanthates (MADIX) polymerization,” *Journal of Polymer Science, Part A: Polymer Chemistry*, vol. 43, no. 22, pp. 5347–5393, 2005.

[21] J. Clayden, N. Greeves, S. Warren, and P. Wothers, *Organic Chemistry*, 2nd ed. Oxford University Press, 2004.

[22] P. J. Flory, “Molecular Size Distribution in Three Dimensional Polymers. I. Gelation,” *Journal of the American Chemical Society*, vol. 63, no. 11, pp. 3083–3090, 1941.

[23] W. H. Stockmayer, “Theory of Molecular Size Distribution and Gel Formation in BranchedChain Polymers,” *The Journal of Chemical Physics*, vol. 11, no. 2, pp. 45–55, 1943. [Online]. Available: http://aip.scitation.org/doi/10.1063/1.1723803

[24] C. D. Vo, J. Rosselgong, S. P. Armes, and N. C. Billingham, “RAFT synthesis of branched acrylic copolymers,” *Macromolecules*, vol. 40, no. 20, pp. 7119–7125, 2007.

[25] A. Matsumoto, Y. Kitaguchi, and O. Sonoda, “Approach to Ideal Network Formation Governed by FloryStockmayer Gelation Theory in Free-Radical Cross-Linking Copolymerization of Styrene with m -Divinylbenzene,” *Macromolecules*, vol. 32, pp. 8336–8339, 1999. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/ma990368k

[26] H. Gao and K. Matyjaszewski, “Synthesis of functional polymers with controlled architecture by CRP of monomers in the presence of cross-linkers: From stars to gels,” *Progress in Polymer Science (Oxford)*, vol. 34, no. 4, pp. 317–350, 2009.

[27] J. McMurry, *Organic Chemistry*, 2008.

[28] A. K. Tripathi, M. L. Neenan, D. C. Sundberg, and J. G. Tsavalas, “Influence of n-alkyl ester groups on efficiency of crosslinking for methacrylate monomers copolymerized with EGDMA: Experiments and Monte Carlo simulations of reaction kinetics and sol-gel structure,” *Polymer (United Kingdom)*, vol. 96, pp. 130–145, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.polymer.2016.04.017

[29] P. Polanowski, J. K. Jeszka, K. Krysiak, and K. Matyjaszewski, “Influence of intramolecular crosslinking on gelation in living copolymerization of monomer and divinyl cross-linker. Monte Carlo simulation studies,” *Polymer (United Kingdom)*, vol. 79, no. December, pp. 171–178, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.polymer.2015.10.018

[30] J. C. Chen and A. S. Kim, “Brownian dynamics, molecular dynamics, and monte carlo modeling of colloidal systems,” *Advances in Colloid and Interface Science*, vol. 112, no. 1-3, pp. 159–173, 2004.

[31] P. Dittrich, J. Ziegler, and W. Banzhaf, “Artificial ChemistriesA Review,” *Artificial Life*, vol. 7, no. 3, pp. 225–275, jul 2001. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/106454601753238636

[32] Z. Li, “DYNAMICS OF POLYMER SELF-ASSEMBLY,” no. May, 2011. [Online]. Available: https://etd.ohiolink.edu/rws_etd/document/get/case1296234501/inline

[33] P. Polanowski and T. Pakula, “Simulation of polymer-polymer interdiffusion using the dynamic lattice liquid model,” *Journal of Chemical Physics*, vol. 120, no. 13, pp. 6306–6311, 2004.

[34] P. Howard, “Modeling with ODE,” pp. 1–26, 2009.

[35] L. Cardelli, “From processes to ODEs by chemistry,” *IFIP International Federation for Information Processing*, vol. 273, pp. 261–281, 2008.

[36] D. T. Gillespie, “A general method for numerically simulating the stochastic time evolution of coupled chemical reactions,” *Journal of Computational Physics*, vol. 22, no. 4, pp. 403–434, 1976.

[37] R. F. T. Stepto, “Dispersity in polymer science (IUPAC Recommendations 2009),” *Pure and Applied Chemistry*, vol. 81, no. 2, pp. 351–353, 2009. [Online]. Available: https://www.degruyter.com/view/j/pac.2009.81.issue-2/pac-rec-08-05-02/pac-rec-08-05-02.xml

[38] M. Schotman, “Size-controlled nanogels by RAFT-polymerization for Biomedical Applications,” Ph.D. dissertation, University of Twente, 2017.

[39] D. A. Shipp and K. Matyjaszewski, “Kinetic analysis of controlled/‘living’ radical polymerizations by simulations. 1. The importance of diffusion-controlled reactions,” *Macromolecules*, vol. 32, no. 9, pp. 2948–2955, 1999.

[40] M. A. Al-Harthi, “Highlight on the Mathematical Modeling of Controlled Free Radical Polymerization,” *International Journal of Polymer Science*, vol. 2015, 2015.

[41] A. K. Tripathi and D. C. Sundberg, “A hybrid algorithm for accurate and efficient Monte Carlo simulations of free-radical polymerization reactions,” *Macromolecular Theory and Simulations*, vol. 24, no. 1, pp. 52–64, 2015.

[42] M. Wulkow, “PREDICI.”

[43] E. Pintos, C. Sarmoria, A. Brandolin, and M. Asteasuain, “Modeling of RAFT Polymerization Processes Using an Efficient Monte Carlo Algorithm in Julia,” *Industrial & Engineering Chemistry Research*, vol. 55, no. 31, pp. 8534–8547, 2016. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/acs.iecr.6b01639

[44] P. Ganjeh-Anzabi, V. Hadadi-Asl, M. Salami-Kaljahi, and H. Roghani-Mamaqani, “A new approach for Monte Carlo simulation of RAFT polymerization,” *Iranian Journal of Chemistry and Chemical Engineering*, vol. 31, no. 3, pp. 75–84, 2012.

[45] M. Drache, G. Schmidt-Naake, M. Buback, and P. Vana, “Modeling RAFT polymerization kinetics via Monte Carlo methods: Cumyl dithiobenzoate mediated methyl acrylate polymerization,” *Polymer*, vol. 46, no. 19 SPEC. ISS., pp. 8483–8493, 2005.

[46] A. Feldermann, A. Ah Toy, H. Phan, M. H. Stenzel, T. P. Davis, and C. Barner-Kowollik, “Reversible addition fragmentation chain transfer copolymerization: Influence of the RAFT process on the copolymer composition,” *Polymer*, vol. 45, no. 12, pp. 3997–4007, 2004.

[47] O. Okay, H. J. Naghash, and I. Capek, “Free-radical crosslinking copolymerization: Effect of cyclization on diffusion-controlled termination at low conversion,” *Polymer*, vol. 36, no. 12, pp. 2413–2419, 1995.

[48] S. Hamzehlou, Y. Reyes, and J. R. Leiza, “A New Insight into the Formation of Polymer Networks: A Kinetic Monte Carlo Simulation of the Cross-Linking Polymerization of S/DVB,” *Macromolecules*, vol. 46, pp. 9064–9073, 2013. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/ma4016054

[49] Y. Zheng, H. Cao, B. Newland, Y. Dong, A. Pandit, and W. Wang, “3D Single Cyclized Polymer Chain Structure from Controlled Polymerization of Multi-Vinyl Monomers : Beyond Flory À Stockmayer Theory,” pp. 13 130–13 137, 2011.

[50] H. Gao, P. Polanowski, and K. Matyjaszewski, “Gelation in living copolymerization of monomer and divinyl cross-linker: Comparison of ATRP experiments with Monte Carlo simulations,” *Macromolecules*, vol. 42, no. 16, pp. 5925–5932, 2009.

[51] F. Rosselló and G. Valiente, “Chemical graphs, chemical reaction graphs, and chemical graph transformation,” *Electronic Notes in Theoretical Computer Science*, vol. 127, no. 1, pp. 157–166, 2005. [Online]. Available: http://dx.doi.org/10.1016/j.entcs.2004.12.033

[52] P. Y. Bruice, *Organic Chemistry*, 7th ed. Pearson, 2014.

[53] D. T. Gillespie, “Exact Stochastic Simulation of coupled chemical reactions,” *The Journal of Physical Chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/j100540a008

[54] T. Schulze, “Efficient kinetic Monte Carlo simulation,” *Journal of Computational Physics*, vol. 227, no. October, pp. 2455–2462, 2008.

[55] S. A. Serebrinsky, “Physical time scale in kinetic Monte Carlo simulations of continuous-time Markov chains,” *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 83, no. 3, pp. 2010–2012, 2011.

[56] A. N. Nikitin, R. A. Hutchinson, M. Buback, and P. Hesse, “Determination of intramolecular chain transfer and midchain radical propagation rate coefficients for butyl acrylate by pulsed laser polymerization,” *Macromolecules*, vol. 40, no. 24, pp. 8631–8641, 2007.

[57] J. Rosselgong and S. P. Armes, “Quantification of intramolecular cyclization in branched copolymers by 1H NMR spectroscopy,” *Macromolecules*, vol. 45, no. 6, pp. 2731–2737, 2012.

[58] T. P. Davis, K. F. O’Driscoll, M. C. Piton, and M. A. Winnik, “Determination of propagation rate constants for the copolymerization of methymethacrylate and styrene using a pulsed laser technique,” *Journal of Polymer Science Part C: Polymer Letters*, vol. 27, no. 6, pp. 181–185, may 1989. [Online]. Available: http://doi.wiley.com/10.1002/pol.1989.140270601

[59] P. M. Fenwick, “A new data structure for cumulative frequency tables,” *Software: Practice and Experience*, vol. 24, no. 3, pp. 327–336, 1994.

[60] S. Halim and F. Halim, “Binary Indexed (Fenwick) Tree,” in *Competitive Programming 3*, 2013, pp. 59–62.

[61] Hackerearth, “Fenwick (Binary Indexed) Trees.” [Online]. Available: https://www.hackerearth.com/practice/data-structures/advanced-data-structures/fenwick-binary-indexed-trees/tutorial/

[62] T. Parr, *The Definitive ANTLR 4 Reference*, 2nd ed., 2013.

[63] A. Unkrig, “JANINO,” 2017. [Online]. Available: http://janino-compiler.github.io/janino/

[64] MDL Information Systems Inc, “CT File Formats,” *Mdl*, no. June, p. 108, 2005. [Online]. Available: http://c4.cabrillo.edu/404/ctfile.pdf

[65] “Protein Data Bank Contents Guide: Atomic Coordinate Entry Format Description,” 2017. [Online]. Available: http://www.wwpdb.org/documentation/file-format-content/format33/v3.3.html

[66] Advanced Chemistry Development Inc., “ACD/ChemSketch User’s Guide.”

[67] J. A. Pomposo, I. Perez-Baena, F. Lo Verso, A. J. Moreno, A. Arbe, and J. Colmenero, “How far are single-chain polymer nanoparticles in solution from the globular state?” *ACS Macro Letters*, vol. 3, no. 8, pp. 767–772, 2014.

[68] M. P. Solf and T. A. Vilgis, “Statistical mechanics of macromolecular networks without replicas,” *Journal of Physics A: Mathematical and General*, vol. 28, no. 23, pp. 6655–6668, dec 1995. [Online]. Available: http://stacks.iop.org/0034-4885/80/i=3/a=036602?key=crossref.60922ac320bf774a09cbce23c9628f9dhttp://stacks.iop.org/0305-4470/28/i=23/a=017?key=crossref.4ee5d7ae7540d72f07e3eec207fb6bc7

[69] H. Tobita, “Dimensions of Cross-Linked Polymers Formed in Living Vinyl/Divinyl Copolymerization,” *Macromolecules*, vol. 27, no. 19, pp. 5413–5420, sep 1994. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/ma00097a022

[70] K. Kajiwara and W. Burchard, “Hydrodynamic radius and dynamic scattering from randomly crosslinked chains,” *Polymer*, vol. 22, no. 12, pp. 1621–1628, 1981.

[71] M. instruments, “Zetasizer Nano Series User Manual,” *Department of Biochemistry Biophysics Facility , University of Chambridge*, no. 2, p. 207, 2004. [Online]. Available: http://www.biophysics.bioc.cam.ac.uk/files/Zetasizer_Nano_user_manual_Man0317-1.1.pdf

[72] International Organization for Standardization, “ISO 22412:2017, Particle size analysis – Dynamic light scattering (DLS),” 2017.

[73] E. Mastan and S. Zhu, “Method of moments: A versatile tool for deterministic modeling of polymerization kinetics,” *European Polymer Journal*, vol. 68, pp. 139–160, 2015.

[74] MOLBASE, “2-butylsulfanylcarbothioylsulfanylpropanoic acid.” [Online]. Available: http://www.molbase.com/en/properties_480436-46-2-moldata-2282757.html

[75] PerkinElmer Informatics, “Chem3D,” 2015. [Online]. Available: https://www.chemsrc.com/en/cas/7098-80-8_767510.html

[76] Sigma-Aldrich, “2,2-Azobis(2-methylpropionitrile).” [Online]. Available: https://www.sigmaaldrich.com/catalog/product/sial/11630

[77] ——, “Solketal methacrylate.” [Online]. Available: https://www.sigmaaldrich.com/catalog/product/aldrich/740950

[78] ——, “Glycidyl methacrylate.” [Online]. Available: https://www.sigmaaldrich.com/catalog/product/aldrich/779342

[79] ——, “Ethylene glycol dimethacrylate.” [Online]. Available: https://www.sigmaaldrich.com/catalog/product/aldrich/335681

[80] National Oceanic and Atmospheric Administration, “Computer-Aided Management of Emergency Operations - Chemical Hazards Response Information System - CHLOROBENZENE,” 1999. [Online]. Available: https://m.cameochemicals.noaa.gov/chris/CRB.pdf

[81] A. G. Oskoei, N. Safaei, and J. Ghasemi, “Densities and Viscosities for Binary and Ternary Mixtures of 1, 4-Dioxane + 1-Hexanol + N , N -Dimethylaniline from T = (283.15 to 343.15) K,” *Journal of Chemical & Engineering Data*, vol. 53, no. 2, pp. 343–349, 2008. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/je700344f

[82] N. Nambu and Y. Sasaki, “Physical and Electrolytic Properties of Monofluorinated Ethyl Acetates and Their Application to Lithium Secondary Batteries,” no. March, pp. 1–9, 2015.

[83] A. Marchetti, C. Pretl, M. Tagllazucchi, L. Tassi, and G. Tosi, “The N,N-Dimethylformamide/Ethane-1,2-diol Solvent System. Density, Viscosity, and Excess Molar Volume at Various Temperatures,” *Journal of Chemical and Engineering Data*, vol. 36, no. 4, pp. 360–365, 1991.

[84] GEO Speciality Chemicals, “Safety Data Sheet for Bisomer® IPGMA,” 2016. [Online]. Available: http://www.geosc.com/Assets/Files/MSDS-Files/BISOMER-IPGMA-(1)/SDS-5008170-BISOMER-IPGMA-KOR-EN.pdf

[85] ChemSrc, “(2,2-dimethyl-1,3-dioxolan-4-yl)methyl 2-methylprop-2-enoate.” [Online]. Available: https://www.chemsrc.com/en/cas/7098-80-8_767510.html

[86] National Center for Biotechnology Information, “PubChem Compound Database - Glycidyl Methacrylate.” [Online]. Available: https://pubchem.ncbi.nlm.nih.gov/compound/7837

[87] ——, “PubChem Compound Database - Glycol Dimethacrylate.” [Online]. Available: https://pubchem.ncbi.nlm.nih.gov/compound/7355

[88] Institut für Arbeitsschutz der Deutschen Gesetzlichen (IFA), “GESTIS Substance Database - 2,2’-Dimethyl-2,2’-azodipropiononitrile.” [Online]. Available: http://gestis.itrust.de

[89] Vesta Intracon B.V., “INIPER 64 (AIBN),” 2015. [Online]. Available: https://vestachem.com/chemicals/azobisisobutyronitrile/

[90] P. J. Flory, “Thermodynamics of high-polymer solutions,” *The Journal of chemical physics*, vol. 1, no. May 2016, pp. 25–30, 1945.

[91] G. Hild, “Interpretation of equilibrium swelling data on model networks using affine and phantom’ network models,” *Science*, vol. 38, no. 13, pp. 3279–3293, 1997.

[92] H. Kato, A. Nakamura, N. Ouchi, and S. Kinugasa, “Determination of bimodal size distribution using dynamic light scattering methods in the submicrometer size range,” *Materials Express*, vol. 6, no. 2, pp. 175–182, 2016. [Online]. Available: http://openurl.ingenta.com/content/xref?genre=article&issn=2158-5849&volume=6&issue=2&spage=175

[93] NanOxiMet, “Particle size and zeta potential analysis via Dynamic Light Scattering (DLS) / Elelctrophoretic Light Scattering (ELS),” pp. 1–7, 2016.

[94] J. C. Moore, “Gel permeation chromatography. I. A new method for molecular weight distribution of high polymers,” *Journal of Polymer Science Part A: General Papers*, vol. 2, no. 2, pp. 835–843, 1964. [Online]. Available: http://doi.wiley.com/10.1002/pol.1964.100020220

[95] H. Chaffey-Millar, D. Stewart, M. M. T. Chakravarty, G. Keller, and C. Barner-Kowollik, “A Parallelised High Performance Monte Carlo Simulation Approach for Complex Polymerisation Kinetics,” *Macromolecular Theory and Simulations*, vol. 16, no. 6, pp. 575–592, 2007. [Online]. Available: http://doi.wiley.com/10.1002/mats.200700028

Appendix A

Polymerization reactions

Figure A.2 contains the list of reactions and molecules the simulation model is based on. Note here that the initiating molecule that is created in reaction 1 is not the same as the one that is released in the RAFT equilibrium in reaction 2. It is also clear that a chiral point is added to the polymer network after each polymerization reaction. This chiral point is indicated by the dashed red circle in reaction 3. As mentioned in Section 2.2.1, a carbon with a free radical electron has a flat structure, but becomes chiral when this free radical becomes part of a covalent bond. The chirality is thus decided when polymerization occurs. For his specific reaction the chances between the left-handed and right-handed orientation are 50-50.

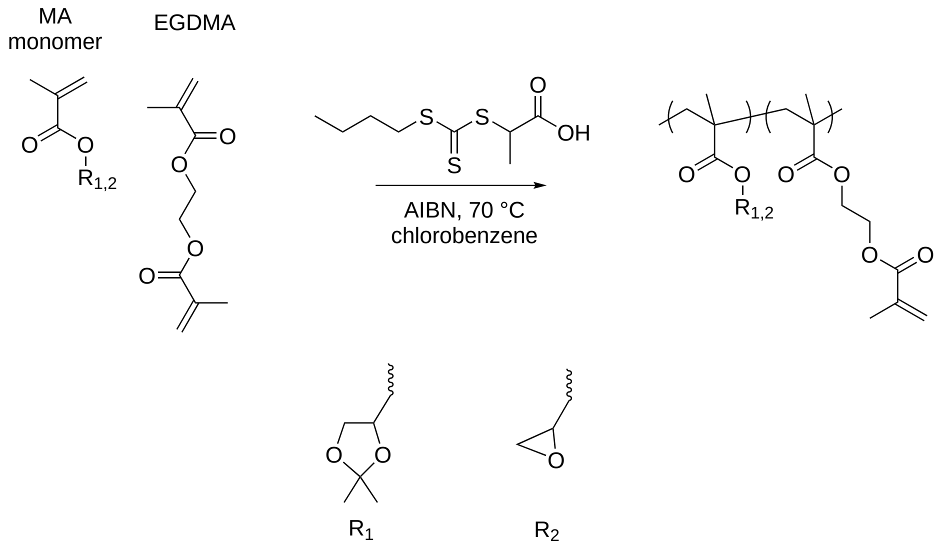


Figure A.1: Overall synthesis of [38]. The rest groups $R_{1,2}$ are those of solketal methacrylate (SMA) and glycidyl methacrylate (GMA) respectively.

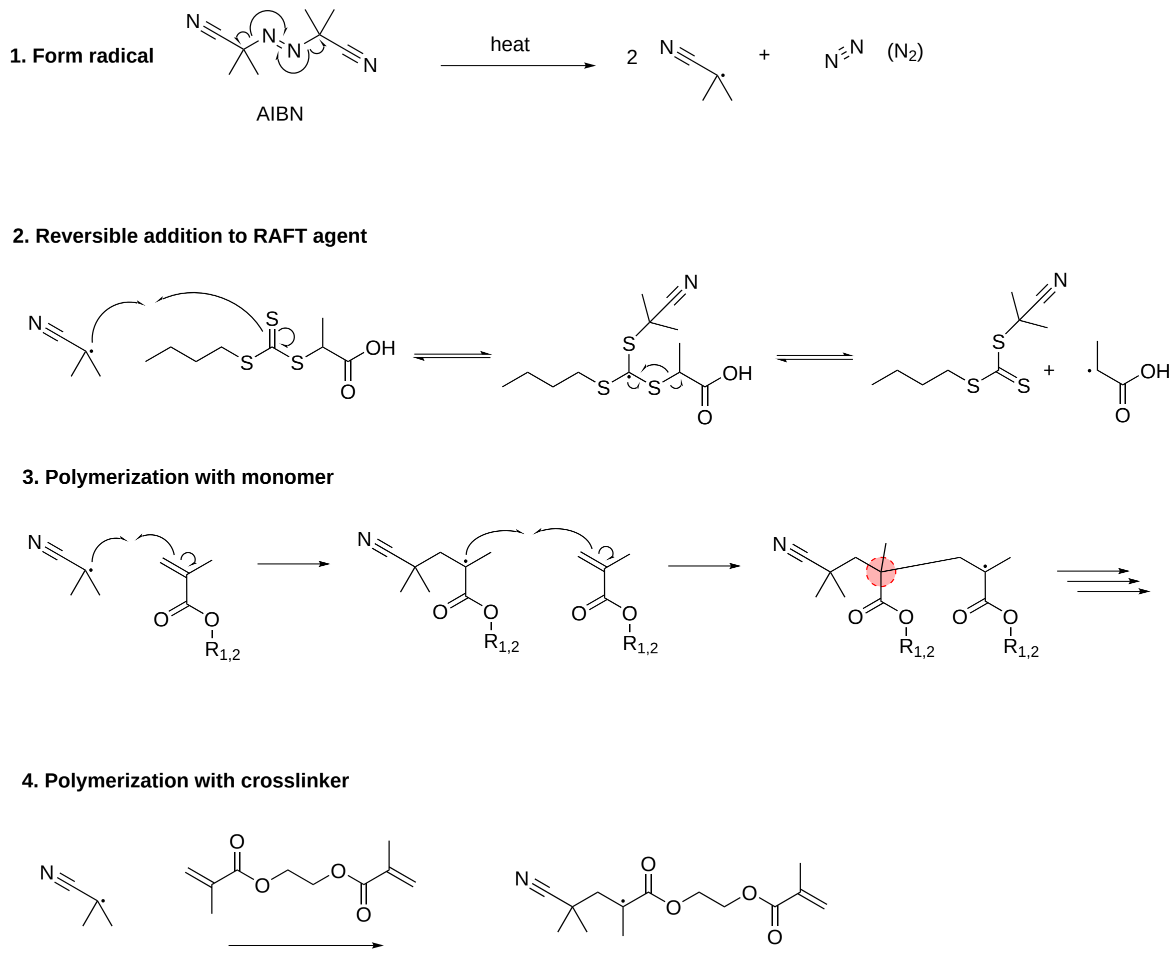

1. Form radical
AIBN
heat
2
(N_2)
2. Reversible addition to RAFT agent
3. Polymerization with monomer
$R_{1,2}$
4. Polymerization with crosslinker

Figure A.2: Chemical reactions that make up the overall synthesis of in Figure A.1. The red dashed circle indicates a chiral point in the emerging polymer chain.

The resonance structures that are relevant are shown in Figure A.3. These structures are relevant for both the monomer's and crosslinker's reactivity, as the reactive double bond of the molecules has a slight positive charge due to it.

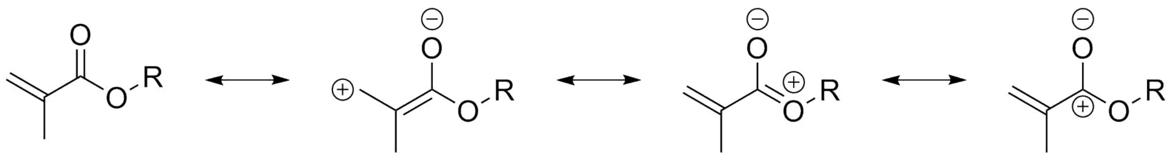


Figure A.3: Resonance structures of monomer and crosslinker

Lastly, the radical-radical reactions that are possible can lead to different products, as shown in Figure A.4 ('Pol' indicates the rest of the polymer network in this figure). Both of the reactions lead to termination however, as is shown in Figure.It is important to differ between these two types of termination, as recombination leads to bigger molecules, while disproportionation does not. One thing to note is that the double bond that is formed in the disproportionation reaction is not reactive and can thus not react with a radical.

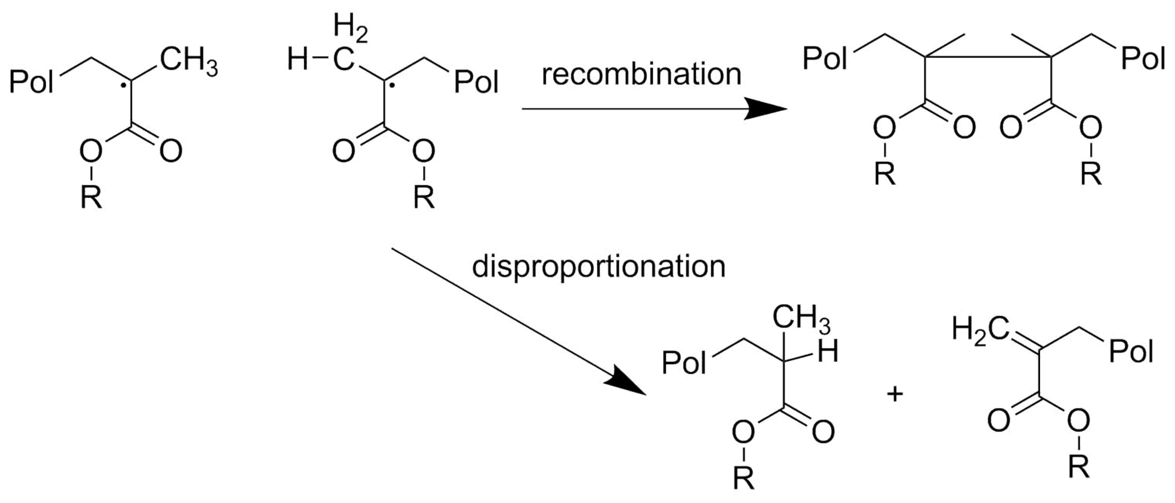


Figure A.4: Radical-radical reactions leading to termination

Appendix B

ANTLR grammar for equations

```
grammar Expression;

expression
     : additiveExpression EOF?
     ;

additiveExpression
     : multiplicativeExpression MINUS additiveExpression
     | multiplicativeExpression ((MINUS|PLUS) additiveExpression)?
     ;//Lookahead necessary to prevent clash with unary minus

multiplicativeExpression
     : powerExpression ((TIMES|DIVISION)? multiplicativeExpression)?
     ;

powerExpression
     : unaryExpression (POWER powerExpression)?
     ;

unaryExpression
     : (MINUS | SQRT)? primaryExpression
     ;

primaryExpression
     : NUMBER
     | VAR
     | PI
     | LPAREN additiveExpression RPAREN
     ;

DIVISION  : '/';
LPAREN    : '(';
MINUS     : '-';
PI        : 'pi'   | '\u03C0';//Unicode symbol for pi
SQRT      : 'sqrt' | '\u221A';//Unicode symbol for square root
PLUS      : '+';
POWER     : '^';
RPAREN    : ')';
TIMES     : '*';
NUMBER    : [0-9]+ ('.' [0-9]+)?;
VAR       : 'EC' | 'EN' | 'MC' | [a-zA-Z];
WS        : [ \t\r\n]+ -> skip ;//Strip whitespace
```

Appendix C

Algorithm extension

This appendix gives a rough outline on how to extend the algorithm presented in Chapter 5. We first describe how to convert our KMC algorithm to a rejection-free KMC. Subsequently, we will explain how non-polymerization reactions like RAFT reactions and initiator reactions can be included in the simulation. Lastly, we show how the reaction rates can be used to further extend the simulation algorithm to also track reaction time.

C.1 Rejection-free KMC

To transform our rejection KMC algorithm into a rejection-free KMC algorithm the reaction selection step has to be changed. Rather than sampling the molecule species and radical position with no regard for steric hindrance and subsequently correcting for it, the steric hindrance factors should be included in the sampling probabilities themselves. This can be done by redefining the total sum of reaction rates. Previously, in Section 4.3 we defined the total sum of reaction rates between initiators,monomers,crosslinkers and polymers as

$$\sum_{Z\in\{EC,EN,MC\}} \sum_{i=1}^{K} c_i r_{Z,i} \left(\frac{v_i f_{Z,i}}{vol_i} + \frac{\sum_{j\neq i}(c_j v_j) f_{Z,j}}{vol_T} + \frac{(c_i-1) v_i f_{Z,i}}{vol_T} \right) \frac{k^{exp}}{A}. \tag{C.1}$$

In Section 5.2.2 we have shown that this equation can be rewritten as

$$\left(\sum_{i=1}^{K} \frac{c_i r_i v_i}{vol_i} + vol_T^{-1} \left(v_T \sum_{i=1}^{K} c_i r_i - \sum_{i=1}^{K} c_i r_i v_i \right) \right) \frac{k^{exp}}{A} \tag{C.2}$$

when omitting the steric hindrance factor $f_{Z,i}$. This factor can be reintroduced by splitting the reaction rates into the same categories used by the steric hindrance factors. We distinguish the following four reaction rates of reactions between:

1. a radical at the end of a polymer chain and a pendent vinyl group, with a steric hindrance factor of ψ
2. a radical at the middle of a polymer chain and a pendent vinyl group, with a steric hindrance factor of ω
3. a radical at the middle of a polymer chain and a vinyl group in a monomer or crosslinker, with a steric hindrance factor of ϕ

4. a radical at the end of a polymer chain and a vinyl group in a monomer or crosslinker, with a steric hindrance factor of 1

By restricting Equation C.2 to each of these categories we get four different equations, which when combined with their steric hindrance factors result into

$$R_{pol} = \left(\begin{array}{r} \psi \left(\sum_{i=1}^{K} \frac{c_i r_{E,i} v_i}{vol_i} + vol_T^{-1} \left((v_T - c_x - 2c_y) \sum_{i=1}^{K} c_i r_{E,i} - \sum_{i=1}^{K} c_i r_{E,i} v_i \right) \right) \\ +\omega \left(\sum_{i=1}^{K} \frac{c_i r_{MC,i} v_i}{vol_i} + vol_T^{-1} \left((v_T - c_x - 2c_y) \sum_{i=1}^{K} c_i r_{MC,i} - \sum_{i=1}^{K} c_i r_{MC,i} v_i \right) \right) \\ +\phi \; vol_T^{-1} \left((c_x + 2c_y) \sum_{i=1}^{K} c_i r_{MC,i} \right) \\ +vol_T^{-1} \left((c_x + 2c_y) \sum_{i=1}^{K} c_i r_{E,i} \right) \end{array} \right) \frac{k^{exp}}{A} \quad \text{(C.3)}$$

where $r_{E,i} = (r_{EN,i} + r_{EC,i})$, and x and y are the indices of molecule species M and C respectively. Note that c_x and $2c_y$ are used to represent the total number of vinyl groups in the monomer and crosslinker species. As shown previously in Chapter 5, such an equation can be used to sample a reaction with the help of Binary Indexed Trees (BITs). In this case BITs should be used to track the following six sums for radical selection with $1 \leq x \leq K$:

1. $\sum_{i=1}^{x} c_i r_{E,i} v_i vol_i^{-1}$
2. $\sum_{i=1}^{x} c_i r_{MC,i} v_i vol_i^{-1}$
3. $\sum_{i=1}^{x} c_i r_{E,i} v_i$
4. $\sum_{i=1}^{x} c_i r_{MC,i} v_i$
5. $\sum_{i=1}^{x} c_i r_{E,i}$
6. $\sum_{i=1}^{x} c_i r_{MC,i}$

The radical selection starts by choosing one of the four categories by sampling the total sum of reaction rates R_{pol}.This should be done in such a way that each addend has the probability of the addend divided by the total sum to be chosen, e.g reactions between chain end radicals and monomers or crosslinkers should be chosen with probability

$$\frac{vol_T^{-1}(c_x + 2c_y) \sum_{i=1}^{K} c_i r_{E,i}}{R_{pol}}$$

Each category also has an associated radical position, either at the end of a polymer chain or in the middle of a polymer chain. In the case of the former we choose a chain end non-crosslinker radical with probability $r_{EN,i} r_{E,i}^{-1}$ and a chain end crosslinker radical otherwise. This way we obtain a value for the radical position Z.

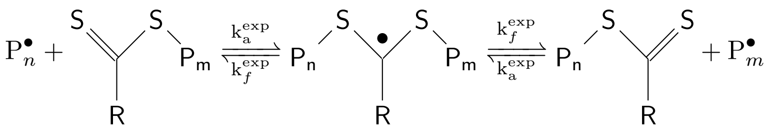
Subsequently, a specific radical is chosen by sampling the respective addend within the equation. The algorithm follows the same steps as the rejection KMC yielding a radical species S_i and either the choice of an inter- or intramolecular reaction.

In the case of an intermolecular reaction a vinyl species should also be chosen. We distinguish two different cases: an intermolecular reaction with a polymer and an intermolecular reaction with a monomer or crosslinker. In the case of the former, the vinyl species is sampled using the BIT that keeps track of $\sum_{j=1}^{K} c_j v_j$, but in contrast to rejection KMC not only the radical species S_i should be excluded, but the monomer species M and the crosslinker species C as well. In the case of the latter we simple sample such that the monomer species M is yielded with probability $c_x(c_x + 2c_y)^{-1}$ and crosslinker species C is yielded otherwise.

The rejection-free KMC allows us to track the sum of reaction rates for polymerization reactions. This, however, is an overestimation, because radicals are not always active, and can thus not always react. By tracking the number of active radicals we can calculate the correct sum of reaction rates. To this purpose RAFT reactions should also be tracked, as the RAFT reactions decide the total number of active radicals. An outline for the inclusion of RAFT reactions is given in the following section.

C.2 RAFT reactions

Another extension to our algorithm would be the explicit inclusion of RAFT reaction, rather than just the effects they cause. There are multiple models available for the RAFT mechanism, but the model that fits our assumptions best is the Slow Fragmentation model [43]. Whereas other models explain the slowdown of the polymerization process by intermediate termination, the Slow Fragmentation model assumes that the slowdown is caused by a small kinetic constant, k_f^{exp}, for fragmentation reactions with the RAFT agent in comparison with a large kinetic constant, k_a^{exp}, for addition reactions with the RAFT agent, i.e.



This leads to a large equilibrium constant k_a^{exp}/k_f^{exp}, meaning that RAFT agents are more often attached to two polymers rather than one.

As all radicals are, on average, active the same amount of time we recommend not tracking which radicals are active, but rather how many. Every RAFT addition reaction lowers this number by one and every RAFT fragmentation reaction increases this number by one. We will refer to RAFT agents attached to two (inactive) radical molecules as RAFT$_2$ and RAFT agents attached to one (inactive) radical molecule as RAFT$_1$. These are considered separate molecule species.

The total reaction rate of addition reactions with a RAFT agent is

$$R_{add} = c_{RAFT_1} \alpha \sum_{i=0}^{K} c_i r_i \frac{k_a^{exp}}{Avol_T}$$

where c_{RAFT_1} is the number of molecules of species $RAFT_1$ and α is the fraction of active reactive centers. The reaction rate of fragmentation reactions is

$$R_{frag} = c_{RAFT_2} 2 k_f^{exp}$$

where c_{RAFT_2} is the number of molecules of species $RAFT_2$.

C.3 Initiator reactions

The last reaction that is not yet simulated is the splitting of the initiator into two initiator radicals. We can simply define the reaction rate of this reaction as

$$R_{init} = c_{Init} k_{init}^{exp}$$

where c_{Init} is the number of initiator molecules in the simulation and k_{init}^{exp} is the experimentally obtained kinetic rate constant. Simulation of this reaction can be done by simply increasing the number of molecules c_I of the initiator radical species by two and reducing c_{Init} by one.

C.4 Time tracking

We can now calculate the total sum of reaction rates in the simulation

$$R_T = R_{add} + R_{frag} + R_{init} + \alpha R_{pol}.$$

We correct the overestimation of the polymerization reaction rates by multiplying this by the previously defined α, since only this fraction of the reactive centers is active. Once the total sum of reaction rates is known, reaction time can also be simulated. The algorithm for this is relatively simple and follows Gillespie's algorithm described in Chapter 3. Time is set to zero at the start of the simulation and is increased by

$$\Delta t = \frac{ln(y^{-1})}{R_T}$$

for every simulated reaction. Here y is again a uniformly distributed random number in the interval $[0, 1)$.

Appendix D

Reaction probabilities

In this chapter we use the notation introduced by Chapter 4. We show that for a given radical molecule species S_i the ratio of intramolecular reaction rates to intermolecular rates, as defined by equation 4.7 through 4.9, is the same as the ratio between the local concentration and global concentration of vinyl groups, when disregarding steric hindrance, i.e. $f_{Z,j} = 1$. Like in Chapter 5 we also use the notations $\overline{R_i}$ and $\overline{R_{i,j}}$ to denote

$$\sum_{Z \in \{EN,EC,MC\}} R_{Z,i}$$

and

$$\sum_{Z \in \{EN,EC,MC\}} R_{Z,i,j}$$

with $f_{Z,j}$ substituted by 1 respectively.

$$\frac{\overline{R_i}}{\sum_{j=1}^{K} \overline{R_{i,j}}} =$$

$$\frac{c_i \dfrac{r_i v_i k^{exp}}{vol_i A}}{-\dfrac{c_i}{vol_T A} r_i v_j k^{exp} + \sum_{j=1}^{K} \dfrac{c_i c_j}{vol_T A} r_i v_j k^{exp}} =$$

$$\frac{\dfrac{c_i r_i k^{exp}}{A} \dfrac{v_i}{vol_i}}{-\dfrac{c_i r_i k^{exp}}{A} \dfrac{v_i}{vol_T} + \dfrac{c_i r_i k^{exp}}{A} \sum_{j=1}^{K} \dfrac{c_j v_j}{vol_T}} = \tag{D.1}$$

$$\frac{v_i vol_i^{-1}}{(-v_i + \sum_{j=1}^{K} c_j v_j) vol_T^{-1}} = \frac{v_i vol_i^{-1}}{(v_T - v_i) vol_T^{-1}} = [L_i][G_i]^{-1}$$

We also show that the ratio between the reaction rate of a single intermolecular reaction between molecules of radical species S_i and vinyl species S_j, $i/neqj$, is the same as the ratio between $c_j v_j$, the total number of vinyl groups in S_j, and $(v_T - v_i)$, the total number of vinyl groups minus the number of vinyl groups in a molecule of species S_i. This is again under the assumption that steric hindrance is disregarded.

$$\begin{aligned}
&\frac{\overline{R_{i,j}}}{\sum_{j=1}^{K}\overline{R_{i,j}}} = \\
&\frac{\frac{c_i c_j}{vol_T A} r_i v_j k^{exp}}{-\frac{c_i}{vol_T A} r_i v_j k^{exp} + \sum_{j=1}^{K} \frac{c_i c_j}{vol_T A} r_i v_j k^{exp}} = \\
&\frac{\frac{c_i r_i k^{exp}}{vol_T A} c_j v_j}{-\frac{c_i r_i k^{exp}}{vol_T A} v_i + \frac{c_i r_i k^{exp}}{vol_T A} \sum_{j=1}^{K} c_j v_j} = \\
&c_j v_j (v_T - v_i)^{-1}
\end{aligned} \tag{D.2}$$

Similarly for a reaction between molecules of radical species S_i and vinyl species S_i we get a ratio of $(c_i - 1)v_j(v_T - v_i)^{-1}$.