8-2020

Application of Software Engineering Principles to Synthetic Biology and Emerging Regulatory Concerns

Justin Firestone
University of Nebraska - Lincoln, justin.w.firestone@gmail.com

Application of Software Engineering Principles to Synthetic Biology and Emerging Regulatory Concerns

by

Justin Firestone

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Doctor of Philosophy

Major: Computer Science

Under the Supervision of Professors Myra B. Cohen and Brittany A. Duncan

Lincoln, Nebraska

August, 2020

Application of Software Engineering Principles to Synthetic Biology and Emerging Regulatory Concerns

Justin Firestone, Ph.D.

University of Nebraska, 2020

Advisers: Myra B. Cohen and Brittany A. Duncan

As the science of synthetic biology matures, engineers have begun to deliver real-world applications which are the beginning of what could radically transform our lives. Recent progress indicates synthetic biology will produce transformative breakthroughs. Examples include: 1) synthesizing chemicals for medicines which are expensive and difficult to produce; 2) producing protein alternatives; 3) altering genomes to combat deadly diseases; 4) killing antibiotic-resistant pathogens; and 5) speeding up vaccine production.

Although synthetic biology promises great benefits, many stakeholders have expressed concerns over safety and security risks from creating biological behavior never seen before in nature. As with any emerging technology, there is the risk of malicious use known as the dual-use problem. The technology is becoming democratized and de-skilled, and people in do-it-yourself communities can tinker with genetic code, similar to how programming has become prevalent through the ease of using macros in spreadsheets. While easy to program, it may be non-trivial to validate novel biological behavior. Nevertheless, we must be able to certify synthetically engineered organisms behave as expected, and be confident they will not harm natural life or the environment.

Synthetic biology is an interdisciplinary engineering domain, and interdisciplinary problems require interdisciplinary solutions. Using an interdisciplinary approach, this dissertation lays foundations for verifying, validating, and certifying safety and security of synthetic biology applications through traditional software engineering concepts about safety, security, and reliability of systems. These techniques can help stakeholders navigate what is currently a confusing regulatory process.

The contributions of this dissertation are: 1) creation of domain-specific patterns to help synthetic biologists develop assurance cases using evidence and arguments to validate safety and security of designs; 2) application of software product lines and feature models to the modular DNA parts of synthetic biology commonly known as BioBricks, making it easier to find safety features during design; 3) a technique for analyzing DNA sequence motifs to help characterize proteins as toxins or non-toxins; 4) a legal investigation regarding what makes regulating synthetic biology challenging; and 5) a repeatable workflow for leveraging safety and security artifacts to develop assurance cases for synthetic biology systems.

ACKNOWLEDGMENTS

It is appropriate to start at the beginning, and thus I first must thank Dr. Jitender Deogun for being the first to encourage me to pursue graduate studies instead of walking a different path. Dr. Deogun's theory classes are thoughtful, challenging, and ultimately inspiring.

I must next thank Dr. Myra Cohen, who recognized my potential as a researcher when I was an undergraduate student in her senior capstone course. Dr. Cohen offered continuous support as her research assistant, and pushed me forward to pursue my own interests even when I struggled to find insights. She wholeheartedly backed my interest in synthetic biology and how it intersects with software engineering. Her guidance and career advice are invaluable, and the sudoku-solver programming assignment from her heuristic search course is still my favorite.

Then I must thank Dr. Brittany Duncan, whose seminar on human-robot interaction led to our successful collaboration on a research paper, for providing timely support when I was unsure how I was going to fund my final semesters. She took a calculated risk by trusting me with a project outside my research focus.

The final member of my committee to thank is Justin (Gus) Hurwitz, who offered one of the nicest compliments after I gave a somewhat controversial presentation at a technology law conference: that I had the "gift of talk." He has encouraged me to continue to explore the intersection of law and technology, which has already resulted in a law journal article. I also thank him for trusting me as a guest lecturer in his domestic cybersecurity course.

Aside from my committee members, I would like to especially thank Mikaela Cashman, whose dedication and zeal are contagious. The success of our collaborations, whether for research or course projects, is largely the result of her energy and pursuit of excellence. I also wish to thank Dr. Massimiliano Pierobon for his encouragement and support as a co-advisor during the first years of my program, as well as an occasional espresso. He was an essential driver for establishing an iGEM team at the University of Nebraska– Lincoln, which allowed me to learn the gritty details of how synthetic biology experiments are conducted in wet labs.

Finally, I must thank my parents, Mark and Sandy Firestone, who have allowed me the freedom to wander around my own way, never getting too worried about my various career choices over the last 20 years. They have supported me in my never-ending thirst for knowledge, and now it’s time for me to use that knowledge for the benefit of others.

GRANT INFORMATION

Funding for this research has been provided in part by the National Institute of Justice Grant 2016-R2-CX-0023, and the National Science Foundation Grant CCF-1901543.

Table of Contents

Preface

- The material from Chapter 3 was published at ASSURE 2018 [70].
- The material from Chapter 4 was published at the Systems and Software Product Line Conference (SPLC) September 2019 [35].
- The material from Chapter 5 was accepted for publication by *Jurimetrics*, the American Bar Association's Section on Science & Technology Law, with an expected publication date of April 2020 [69].

List of Figures

List of Tables

Chapter 1

Introduction

Synthetic biology (SB) is the practice of engineering living organisms by modifying their DNA. The discipline of SB has advanced rapidly over the last 30 years due to overcoming key technical challenges such as repeatability of experiments and standardization of parts [33]. Recent successes include: sensing heavy metals for pollution mitigation [19]; development of synthetic biofuels [226]; engineering cells to communicate and produce bodily tissues [181]; emerging medical applications [118, 221]; and basic computational purposes [47]. This dissertation uses the acronym “SEBO” to stand for “Synthetically Engineered Biological Organisms.”

Synthetic biologists design new functionality, encode it in DNA strands, and insert this new DNA component into a living organism such as the common K-12 strain of the bacteria *Escherichia coli (E. coli).* As the SEBO reproduces, it replicates the new DNA along with its native DNA code and builds proteins that perform an engineered functionality. Software engineers program computing systems to perform new functions, and synthetic biologists program SEBOs to perform new functions. Programming new cellular behavior is becoming easier and more prevalent as companies offer increasingly affordable DNA synthesis and design services.

A DNA sequence is a string of four base pairs A (adenine), C (cytosine), G (guanine), and T (thymine) which can be combined in any order [11]. It is possible to encode “logic” within cells using different parts to build molecular communication systems, either intracellular or intercellular [11]. Engineers can design cells which, similar to silicon devices, can sense, actuate, compute, and communicate based on the flow of molecular signals at the nanoscale [3]. SEBOs can thus be described as living computational systems running software. Biological software, like traditional software, needs to be tested, validated, and verified for correctness within its run-time environment. A key insight of this dissertation is that traditional software engineering principles could be beneficial for testing, validating, and verifying SEBOs.

Furthermore, biological software, like traditional software, could be designed negligently or with malicious intent. This means SB is a “dual-use” technology because its potential for beneficial or malicious uses. Researchers have already created dangerous SEBOs, such as: 1) building the polio virus in 2002 [191]; 2) reconstructing the Spanish Flu virus in 2005 [209]; 3) booting a novel bacteria DNA genome in 2010 [87]; 4) engineering a strain of the Avian Flu virus to be more communicable among mammals in 2012 [105]; 5) and making a new strain of the Spanish Flu virus in 2014 [63]. Synthetic, home-brewed heroin was considered feasible back in 2014 [59, 219]. In 2019, a gene-editing company unintentionally inserted a bacterial plasmid into a bull, fortunately with no obvious negative consequences [152].

More people are engaging in SB every year, with the notable example of the International Genetically Engineered Machine (iGEM) Competition held annually at MIT in Boston, Massachusetts [102]. Over 300 teams of students, from high school to graduate, compete by designing SEBOs to solve real-world problems. Teams must submit their engineered parts along with their designs and experimental results to an open-source collection of genetic components called “BioBricks.” This BioBrick repository, called The Registry of Standard Parts [265], contains over 45,000 DNA sequences with descriptions in various detail. In fact, the SB community aggressively encourages an open-source approach to democratize and “de-skill” the technology [63], and affordable desktop SB kits are available for sale online [202].

The technology of SB is rapidly changing and current U.S. laws and international regulatory frameworks are ill-suited for such a rapidly moving target. It remains unclear which U.S. agencies should have jurisdiction to regulate SEBOs, let alone what research activities should be prohibited. These issues are of extreme importance, because engineers are creating SEBOs with new functions and little to nothing is known about how these new organisms will react in the environment or affect “natural” life [267].

To address these issues, this dissertation explores the following topics:

- The application of assurance cases and assurance patterns to support safety and security of SB designs;
- The application of feature models to SB designs to more easily find parts to meet safety and security requirements;
- A discussion of what makes SB difficult to regulate on both a domestic and international level, with suggestions for collaborations among synthetic biologists, software engineers, and cybersecurity experts;
- An extension of prior work to analyze motifs in DNA sequences for characterizing them as toxins or non-toxins, along with new ways to visualize motif data; and
- A new workflow process for leveraging fault trees and attack-defense trees to semi-automatically generate assurance cases for SB designs.

1.1 Motivating Example

Imagine that the hottest new consumer product of next year is the “Chianettle,” an Internet of Things (IoT) biological system which lives and grows in an aquarium connected to a tablet or personal computer, a high-level overview of which is depicted in Figure 1.1. The organism is the product of SB, a “smart” algae which users can interact with through a microfluidic chamber. As seen in Figure 1.2, the microfluidic chamber has sensors and actuators which provide real-time feedback through a mobile application, and users can either manually intervene or opt for automatic mode, which will adjust the aquarium’s environment to create ideal conditions with respect to temperature, pH level, nutrient level, as well as frequency and duration of light.[1]

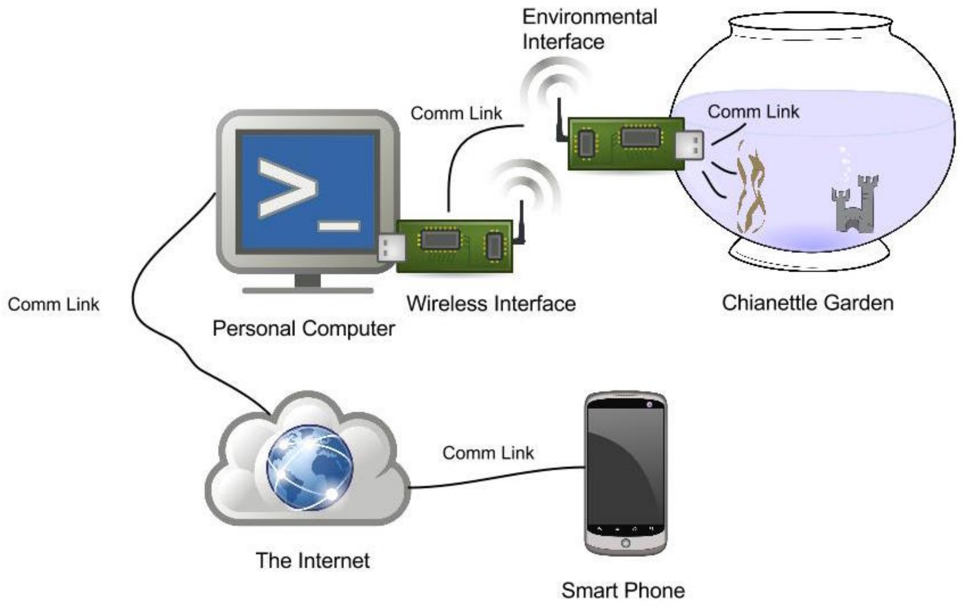


Figure 1.1: A high-level overview of the “Chianettle,” an imaginary SB product.

What makes Chianettle truly special is that it is the first “social media” plant. Users can link their individual plants to various online services such as Facebook,

[1]The Chianettle is partially inspired by: 1) the DIY Glowing Plant Kickstarter project [240], which is no longer active; 2) recent developments using SB to engineer plant-to-plant communication [231], and 3) progress in designing various switches and Boolean logic gates in plants [6].

Instagram, and Snapchat, allowing friends to become part of the story and track the progress of other plants, even plants linked to celebrity and influencer accounts. There is also a competition ladder where users from across the globe challenge each other to make their Chianettles the best in the world in various characteristics such as size, growth rate, or luminosity.

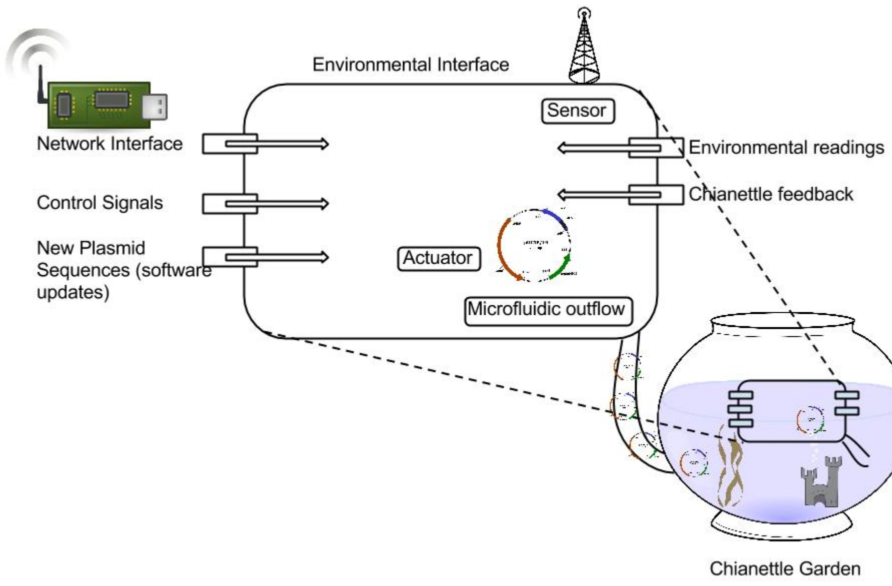


Figure 1.2: A low-level overview of the "Chianettle," an imaginary SB product.

For a monthly subscription fee, users can access a library of pre-approved and well-documented DNA plasmids specifically designed for altering the behavior and characteristics of their plants. Users can either order the plasmids via mail or synthesize them at home using optional add-on hardware, the Chianettle Plasmid Printer. In addition to the subscription service, there exists a vigorous open-source DIY community devoted to sharing custom plasmids which enable the algae to behave in ways never seen before in nature. Finally, there is an unofficial mobile application developed and maintained by the DIY community which offers more automation than the manufacturer's software, allowing users to "one-click" a plasmid from the open-source database, download the DNA sequence to the Plasmid Printer, synthesize the plasmid, and then send it to the microfluidic chamber to transform the algae with a new genetic program.

Now for the bad news. It is unclear what will happen if a Chianettle escapes its aquarium, such as being dumped into a toilet, swimming pool, or lake. In addition, as an IoT device, all of the software and hardware interfaces have publicly-facing network connections making them vulnerable to hacking. It is not clear what new behaviors can be introduced by DIY tinkering, let alone malicious hacking.

As an emerging engineering discipline, SB offers a plethora of multifaceted, interdisciplinary challenges. The average length of a single plasmid can be 5,000bp, or $4^{5,000}$ possible sequences, which is such a vast configuration space that it would be impossible to exhaustively test the full configuration space before the theoretical end of the universe. Engineered SB organisms are also safety-critical systems, because they have the potential to cause harm to life or the environment.

A core theme of this dissertation is that if SB is programming life through manipulating DNA, then traditional computer science concepts and software engineering principles could apply to SB and help create interdisciplinary solutions to interdisciplinary problems. Perhaps feature models from software product lines engineering could help synthetic biologists reason about the vast configuration space and suggest pruning, sampling, and prioritization for wet-lab experiments (see Chapter 4). Building assurance cases for safety-critical domains such as avionics or nuclear energy could help synthetic biologists contemplate safety and security features earlier in the design-build-test cycle and ultimately provide evidence to support validation and verification of their new biological systems (see Chapter 3). From a domestic and international regulatory viewpoint, computer scientists and cybersecurity experts could help synthetic biologists gain better awareness of the weaknesses and capabilities of software and hardware (see Chapter 5).

The design-build-test cycle comprises several different software and hardware systems, all of which represent multiple attack surfaces for malicious actors to exploit. Concepts from text-compression algorithms could help distinguish which DNA sequences are toxic or non-toxic, including new ways to visualize DNA motifs (see Chapter 6). Lastly, concepts of threat modeling and fault-tree analysis could improve the security and integrity of the entire SB process, including physical threats to facilities and the supply chain (see Chapter 7).

1.2 Contributions and Overview

This dissertation lays some of the foundation for validating and verifying the safety and security of SEBOs. Although many SEBOs will be intended for use only within production facilities, many will be intended for release for application within the general environment. Traditional software engineering concepts about safety, security, and reliability of systems could inform synthetic biologists about how to incorporate similar concepts in their SEBO designs, hopefully early on in the development process. It is possible to develop feature models, fault trees, attack-defense trees, and assurance cases for SB designs. These concepts can also help stakeholders navigate what is currently a confusing regulatory process.

Law and technology operate at very different paces. Technology frequently advances so rapidly that laws enacted just a few years ago can seem ineffective or short-sighted. If it is appropriate to apply software-engineering principles to SB, then those principles could help lay foundations for a more pragmatic governance structure, both domestically and internationally.

As with any technology, SB has a dual-use nature, and it is important that regulations aimed at SB effectively protect the public from malicious or negligent activity without stifling legitimate, beneficial innovation. This dissertation provides guidance and hopefully inspires greater interdisciplinary collaboration between synthetic biologists and software engineers to address the very difficult challenges related to designing, testing, building, validating, verifying, and certifying SB organisms and their outputs.

Chapter 2 provides the general information about synthetic biology needed to situate the rest of the chapters. Chapter 3 demonstrates how a traditional software-engineering principle of an assurance case could apply to SEBOs. An assurance case is a document which depicts safety and reliability architectures in a tree-like graph, using natural language to meet specific goals supported by strategies and evidence. Chapter 3 adds the concept of a sub-pattern called a “recipe” to assist synthetic biologists with their construction. Typically, a synthetic biologist would have no experience with or knowledge of assurance cases, and the core idea is to use SB-specific domain knowledge to reduce the degrees of freedom available when constructing an assurance case, as well as providing specific guidance about what language is appropriate for filling in individual nodes.

Chapter 4 demonstrates how the traditional software-engineering concept of feature models can be used to reason about the highly configurable nature of synthetic biology as programming life. A feature model is an abstract representation of core software modules used to show all of the possible software products which could be developed, and all of those products comprise a software product line. Feature models are depicted as trees which will often include cross-tree restraints and dependencies. The intuition is that open-source DNA parts repositories have much in common with open-source software repositories, and both can be represented as a software product line using feature models. It should then be possible, through application of SB domain knowledge, to build a feature model for SEBOs with respect to desired characteristics such as cell-to-cell communication systems or safety features such as kill switches.

Chapter 5 explores the difficulties of regulating SB from a nation-state perspective and suggests increased awareness and cooperation among all stakeholders. Although there were several attempts for international cooperation to regulate SB 10-15 years ago, today there exist few formal legal requirements to ensure the safety and security of SB projects, whether it be at the industry, academic, or do-it-yourself (DIY) level.

Chapter 6 presents a tool which extends upon related work to analyze DNA sequences for detecting whether proteins are toxic or non-toxic. The tool analyzes the frequencies of DNA motifs within larger sequences and provides some new visualizations for this type of data. Chapter 7 suggests a new process for research, industry, and government to leverage pre-existing safety and security artifacts such as fault trees and attack-defense trees to semi-automatically generate assurance cases for novel SB applications. The hope is that the process of Chapter 7 will address some of the concerns raised in Chapter 5. Lastly, Chapter 8 offers some areas for future research and a brief conclusion.

Chapter 2

Background

2.1 Synthetic Biology Basics

There is still no universally agreed upon definition of SB, but generally it is an engineering discipline which emphasizes “the design and construction of new biological entities such as enzymes, genetic circuits, and cells or the redesign of existing biological systems.” [246] Through SB techniques, it is possible to manipulate DNA sequences to create entirely new organisms with behavior and characteristics never seen before in nature [12].

The ability to synthesize DNA sequences in any order and length offers seemingly limitless potential for creating biological systems, similar to how programming computers is ultimately engineering zeros and ones in any order and length passed through a processing unit. Through SB, an engineer can code the nucleotide bases adenine (A), cytosine (C), guanine (G), and thymine (T) in a sequence which can be read and interpreted by living cells to trigger specific behavior. Although SB designs are theoretically limited only by human creativity, the most basic SB stand-alone functional unit is called a transcriptional unit [12]. Figure 2.1 shows an abstraction of a transcriptional unit using Synthetic Biology Open Language (SBOL), [280] currently the most popular standard for describing SB components for sharing and reuse [80].

The transcriptional unit is a composite part composed of four basic DNA

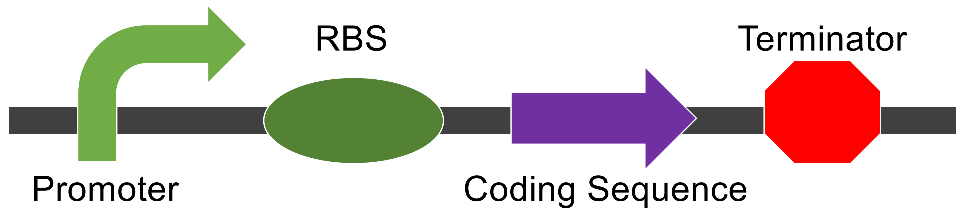


Figure 2.1: SBOL model of a transcription unit.

parts. The promoter, also called a regulator, recruits cellular components to initiate transcription which translates DNA into mRNA. A promoter is sometimes referred to as a regulator because it can also inhibit transcription and act like an inverse gate. Promoters can be either constitutive or inducible. A constitutive promoter encourages transcription without any trigger, whereas an inducible promoter can be triggered on or off depending upon the presence or absence of specific proteins or other environmental conditions. The RBS marks the binding location of the ribosome, which is the organic machine that translates the mRNA. The coding sequence represents the desired protein to be produced. The terminator marks the end of transcription of the protein coding sequence.

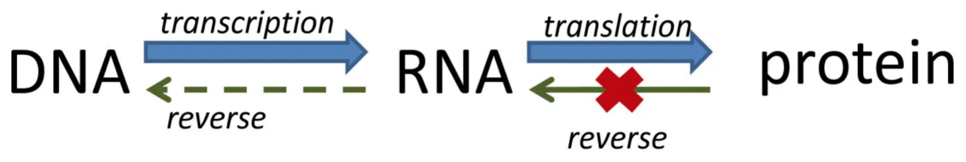


Figure 2.2: Information flow according to the central dogma of molecular biology [121].

This is the basis for the “central dogma” of the flow of information in molecular biology: DNA is transcribed into RNA and RNA is translated into proteins [80]. Figure 2.2 shows that the transcription process can be bi-directional, but it is believed that reverse translation, or that a cell could convert a protein back into RNA, is not possible based on current knowledge of cellular processes [121].

Typically, the transcriptional unit is synthesized into a plasmid, depicted as a ring like the one shown in Figure 2.3. The map can include information about the subsequences required to build the plasmid, along with their individual functions.

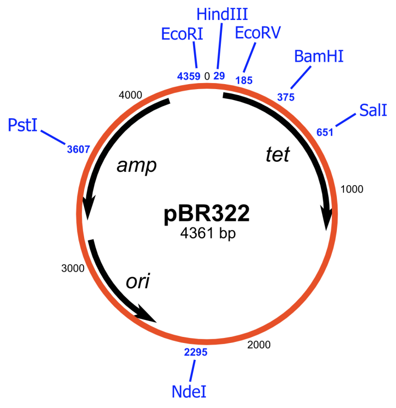


Figure 2.3: Example plasmid map.

In a biological wet lab, an engineer can transform existing cells by inserting copies of the plasmids. Experiments could involve any type of cells, but the most commonly used cells are the K-12 strain of *E. coli*, which is considered to be generally safe if exposed to healthy adult humans [268]. Once transformed, the cells interpret the DNA of the plasmid as if it were its own native DNA. The cells will use their own resources to execute the new program by transcribing the plasmid DNA into RNA and translating the RNA into proteins which the cells would not otherwise produce naturally [80].

The ability to make a cell produce a novel protein with a single plasmid, by itself, offers a theoretically vast number of designs, because a plasmid is often 5,000bp in length. With an alphabet of four nucleotide bases, that means there are $4^{5,000}$ or $2^{10,000}$ different ways to make even the most basic SB unit. One one hand, $4^{5,000}$ is an over-approximation because not every possible sequence of 5,000 DNA nucleotides will represent a meaningful message for cells to interpret. On the other hand, it is an under-approximation because some cells can be transformed with plasmids of 10,000bp or longer, and a design can include multiple plasmids or plasmids which have multiple functions chained together [80], including engineered communications with other types of cells in a sender-receiver-response system [220].

2.2 DNA Sequencing and Synthesis

Although any sequence can be synthesized, theoretically even randomly, SB designs use parts which have been sequenced from nature. In 1977, Frederick Sanger and colleagues introduced the first breakthrough technique for sequencing DNA [189]. Sanger sequencing is expensive and slow, but an accurate process. In the 1990s, J. Craig Venter and colleagues came up with the idea of “shotgun” sequencing. Venter’s method could handle longer sequences more quickly by chopping them up for parallel Sanger sequencing [80]. By 2008, with the addition of high-speed computing and larger RAM capacities, next-generation sequencing (NGS) emerged as the dominant technique [80].

In NGS, copies of longer sequences are generated through a process called polymerase chain reaction (PCR) and then chopped up into shorter chunks of roughly 100bp. A dedicated machine reads the chunks, with each of the four different base pairs emitting a different frequency of light from a chemical dye [189]. The reads, numbering in the hundreds of thousands, must then be aligned back together into the original sequence. The alignment process requires large amounts of RAM, but is quicker than traditional Sanger sequencing [130]. The improved speed of NGS has a tradeoff, however, of a higher error rate [233].

Once aligned, the output is simply a text file in the FASTA format shown in Figure 6.2. The major open-source database for sequenced proteins is called UniProt [7], where users can search over 160 million entries for proteins sequenced from nature, such as those found in viruses, bacteria, plants, and animals. Many sequences have additional information like protein structure or characteristics.

An engineer can use a database like UniProt to find a protein with a desired

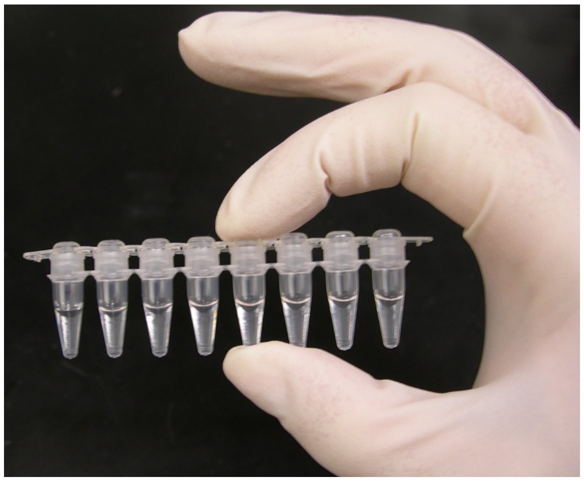


Figure 2.4: Example of DNA test tubes.

function, but more information is needed to build a plasmid that will function inside a transformed cell. Figure 2.1 shows that, at a minimum, the plasmid also needs a promoter, an RBS, and a terminator. As an engineering discipline, SB emphasizes reuse, modularity, and composability of parts. This has led to another type of open-source database, a DNA parts database with annotated information about specific biological functions and compatibility of each sequence [265]. The DNA parts are often called “BioBricks,” a standardization concept introduced in 2003 by Tom Knight at MIT [120]. A BioBrick can theoretically be assembled in any order, copied and pasted into a text file, and uploaded to a DNA synthesis company which will use an automated manufacturing process to chemically build the desired plasmid at a cost of less than $0.10 per base pair [80].

The company will ship an amount of the plasmid known as an aliquot in test tubes like those shown in Figure 2.4. Then the engineer can create more copies of the plasmid via PCR, also known as amplification. The last step is to choose a host organism for transformation, a process which stresses cellular membranes to open pores large enough to receive the plasmid, after which the cells will begin to process the new DNA as if it were its own [80].

2.3 The iGEM Competition

As an example of how quickly SB is growing, the iGEM competition was first held in the summer of 2004 at MIT with five U.S. teams, and the 2018 competition hosted 340 teams from 42 countries, with more than 5,000 students participating [249]. iGEM describes itself as a competition where teams “design, build, test, and measure a system of their own design using interchangeable biological parts and standard molecular biology techniques.” [259] Students design projects which often address various regional problems such as pollution mitigation. Teams are judged by community experts and can be awarded a medal (bronze, silver, and gold) corresponding to the impact and contributions of their project. A gold medal team must achieve several goals such as modeling their project, demonstrating their work through experimentation, collaborating with other teams, addressing safety concerns, improving pre-existing parts or projects, and contributing new parts to the BioBrick database.

Teams work through the summer to complete their projects, some of which are “quite sophisticated.” [156] The competition emphasizes collaborations with other teams, and expects all participants to follow safety guidelines [269]. Work must be done in labs which are certified to be safe for handling the equipment, materials, and organisms [268].

It’s not just 5,000 students every year, of course, who are engaged in SB. Billions of dollars are invested each year in private SB companies, and there are individuals who engage in SB experimentation in their garages [156], an example of which can be seen in Figure 5.1. The technology is moving rapidly, becoming cheaper and easier to use.

This raises some concerns, because if anyone with $20,000 and a garage can program life, then we will see “good” programs, “bad” programs, and “buggy” programs, regardless of programmer intent.

2.4 Software Engineering Concepts Applied to Synthetic Biology

More specific details are provided in the subsequent chapters, but this section presents a brief explanation of the software engineering concepts used in this dissertation.

2.4.1 Assurance Cases

Assurance cases, sometimes referred to as safety cases, are tree-like graphs which feature nodes filled with natural-language text to support what should be a “clear, comprehensive, and defensible argument that a system is acceptably safe to operate in a particular context.” [116] Assurance cases are commonly accepted and expected for safety-critical systems in Europe, where they were originally invented in response to several deadly offshore drilling and railway accidents in Britain during the late 1980s and early 1990s [116]. Since then, assurance cases have found acceptance in other safety-critical domains such as avionics [51], nuclear power [50], medical devices [110, 112, 136], and sub-sea engineering [234].

Figure 2.5 shows what kind of text each type of node should contain, along with the hierarchical structure of safety goals being supported by strategies, sub-goals, and evidence. Chapters 3 and 7 provide more details about assurance cases and their uses.

2.4.2 Feature Models

Feature models are also tree-like graphs which help software engineers reason about the variability and commonality of software components within a large code base, which is called a software product line [21]. The configuration space of a

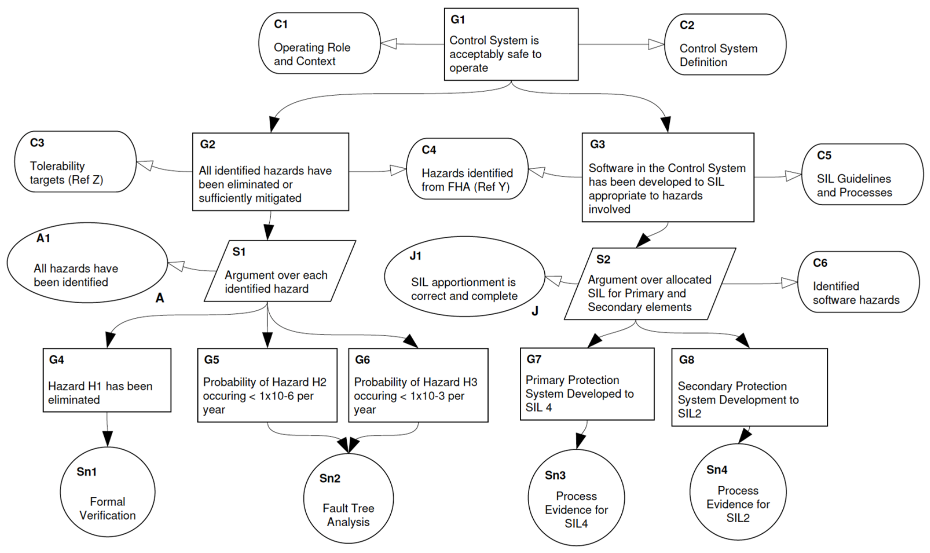

C1
Operating Role and Context
G1
Control System is acceptably safe to operate
C2
Control System Definition
C3
Tolerability targets (Ref Z)
G2
All identified hazards have been eliminated or sufficiently mitigated
C4
Hazards identified from FHA (Ref Y)
G3
Software in the Control System has been developed to SIL appropriate to hazards involved
C5
SIL Guidelines and Processes
A1
All hazards have been identified
A
S1
Argument over each identified hazard
J1
SIL apportionment is correct and complete
J
S2
Argument over allocated SIL for Primary and Secondary elements
C6
Identified software hazards
G4
Hazard H1 has been eliminated
G5
Probability of Hazard H2 occuring < 1x10-6 per year
G6
Probability of Hazard H3 occuring < 1x10-3 per year
G7
Primary Protection System Developed to SIL 4
G8
Secondary Protection System Development to SIL2
Sn1
Formal Verification
Sn2
Fault Tree Analysis
Sn3
Process Evidence for SIL4
Sn4
Process Evidence for SIL2

Figure 2.5: A generic assurance case using Goal Structuring Notation [184].

large code base typically grows quite large. As an example, one code base of 250 features allowed roughly 2^{105} valid products [21].

Feature models have been applied to commercial software production to handle mass customization of consumer products for embedded systems, mobile devices, automobiles, and avionics [17, 216]. They have also been applied to open-source projects such as the Linux kernel [195] and GNU Compiler Collection [85].

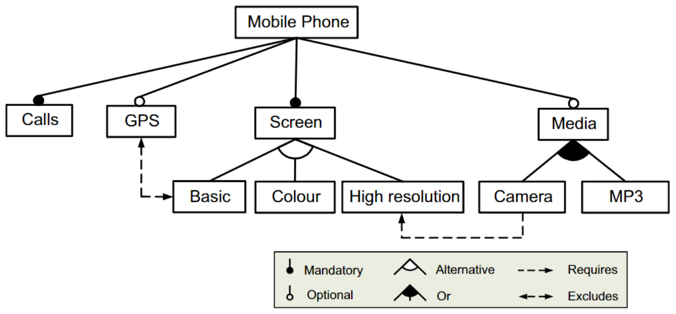


Figure 2.6: A generic feature model with legend [17].

Feature models help software engineers visually comprehend vast configuration spaces by categorizing features as optional or mandatory, and whether zero, one, or many of a feature can be included to create a valid software product. There can also be cross-tree constraints. Figure 2.6 shows a generic feature model for a mobile phone and how to draw relationships between nodes. If GPS is chosen as an option, then choosing a basic screen is prohibited by a cross-tree constraint. If a camera is included, a high-resolution screen is required.

The structure of feature models allows software engineers to calculate how many products can be generated from a large code base. Importantly, it allows a determination of which products are valid, redundant, or violate constraints [21]. Chapter 4 provides further background on feature models.

2.4.3 Fault Trees and Attack-Defense Trees

Fault trees have been used since at least the 1970s to analyze the different ways safety-critical software could enter failure modes [76]. The leaves of the fault tree represent individual faults which should be mitigated to ensure a proper level of safety. Nodes can have AND or OR relationships, meaning some failure modes trigger based on two or more faults, while others trigger only upon the confluence of two or more events. Fault trees can also assign probabilities and severities to events as part of the analysis which allows engineers establish priorities for mitigation strategies [179].

Figure 2.7 shows a small fault tree for an insulin pump. Unintentional administration of insulin (F1) can occur either because of an incorrect command from the controller (F2), or because an excess bolus of insulin was administered (F3). The excess bolus can be caused by too many requests from the user (F4) only if the safeguard against excessive requests fails (F5).

Attack-defense trees look to verify system security, and usually assume an intelligent actor as an adversary [187]. Attack nodes, typically in red, represent the various ways an adversary could compromise the system security, whether

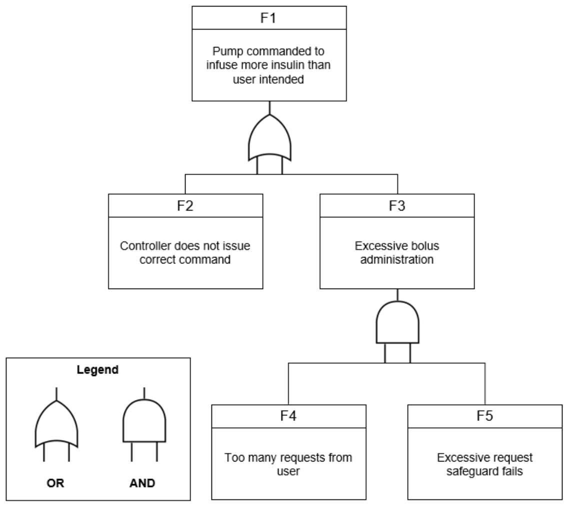

F1
Pump commanded to infuse more insulin than user intended
F2
Controller does not issue correct command
F3
Excessive bolus administration
F4
Too many requests from user
F5
Excessive request safeguard fails
Legend
OR
AND

Figure 2.7: An example fault tree based on [111].

physically or logically. A basic example for compromising a network server is shown in Figure 2.8.

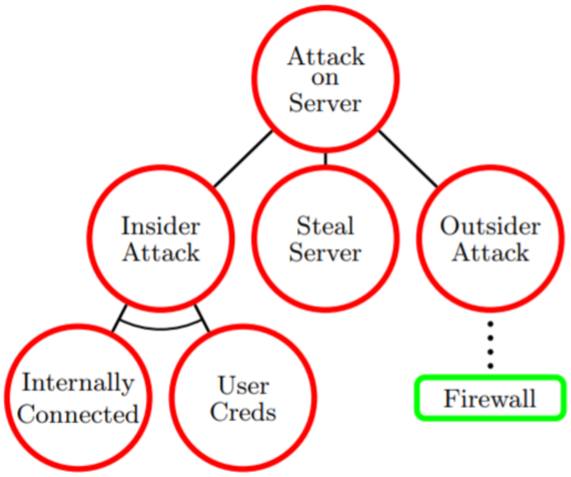


Figure 2.8: An example attack-defense tree [123].

The three nodes under the root node of Attack on Server indicate alternative, independent attacks. The arc over the two nodes under Insider Attack indicate both nodes must occur to accomplish that attack. Ideally, the leaves of the tree will be mitigation nodes, typically in green, appended directly below the attack leaves. As with fault trees, attack nodes can include quantitative aspects such as probabilities and severities for prioritizing mitigation strategies [122]. More information about fault trees and attack-defense trees appears in Chapter 7.

Chapter 3

Facilitating Assurance Cases Through Domain-Specific Patterns

Assurance cases have grown in popularity to reason about safety-critical systems. They are commonly used in domains such as aviation, nuclear energy, railways, and offshore drilling [22, 68, 117, 197]. Recent work has proposed using assurance cases in non-traditional domains where engineering safety is also paramount such as SB and medical devices [44, 110]. The discipline of SB is safety-critical because the SEBOs and their outputs have the potential to cause harm to life or the environment, leading to the need to assure their safety. Recent work has suggested assurance cases are appropriate for this purpose [44].

An assurance case is built in a hierarchical tree structure, with the root node being the main claim of safety supported below by arguments that are supported by evidence [22]. Claims are “assertions put forward for general acceptance” and are “typically statements about a property of the system or some subsystem.” [22] Evidence is “used as the basis of the justification of the claim” and “may include the design, the development process, prior field experience, testing, source code analysis or formal analysis.” [22] Arguments “link the evidence to the claim” along with “validation for the scientific and engineering laws used.” [22]

A generic example is shown in Figure 3.1 and provides a reference key for Goal Structuring Notation (GSN), the most common standard for depicting

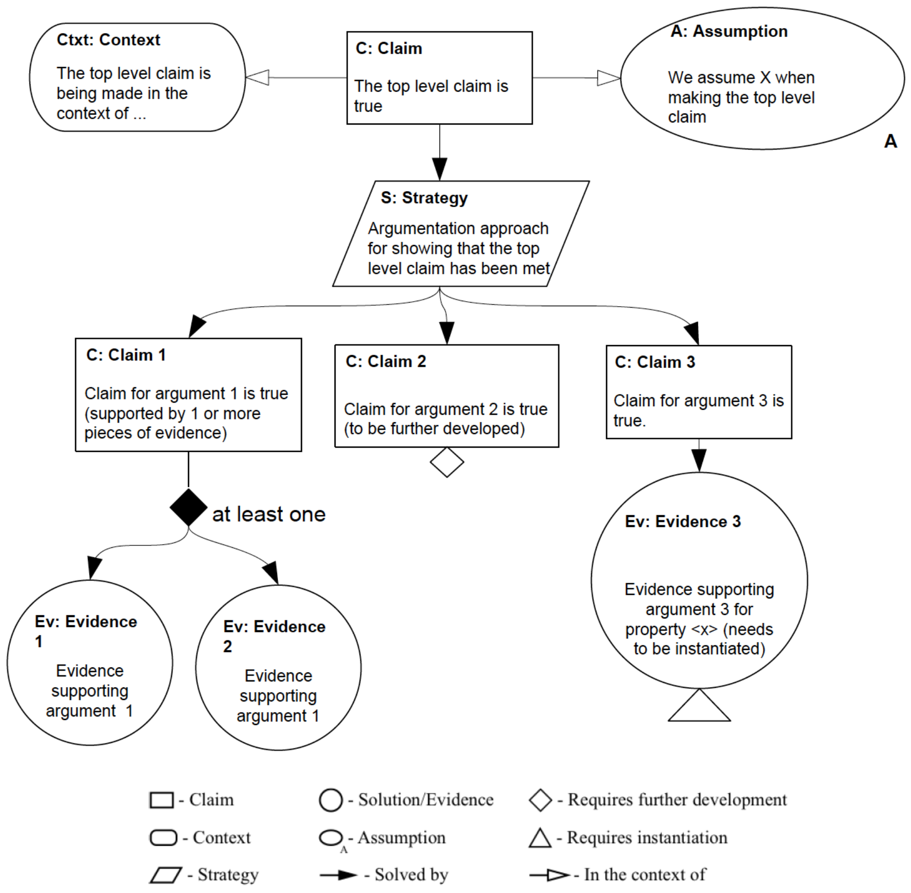

Ctxt: Context
The top level claim is being made in the context of ...
C: Claim
The top level claim is true
A: Assumption
We assume X when making the top level claim
A
S: Strategy
Argumentation approach for showing that the top level claim has been met
C: Claim 1
Claim for argument 1 is true (supported by 1 or more pieces of evidence)
C: Claim 2
Claim for argument 2 is true (to be further developed)
C: Claim 3
Claim for argument 3 is true.
at least one
Ev: Evidence 1
Evidence supporting argument 1
Ev: Evidence 2
Evidence supporting argument 1
Ev: Evidence 3
Evidence supporting argument 3 for property <x> (needs to be instantiated)
- Claim
- Context
- Strategy
- Solution/Evidence
- Assumption
- Solved by
- Requires further development
- Requires instantiation
- In the context of

Figure 3.1: A generic assurance case using GSN notation [223].

assurance cases [223]. Claims are rectangles labeled with the letter “C,” and the primary claim is the root node. Assumptions about those claims are drawn with ovals labeled with an “A.” If claims have specific contexts or environments, they are rectangles with rounded corners with “Ctxt” for a label. Finally, strategies are parallelograms labeled with an “S” and serve as the basis for arguments supported by circles of evidence labeled with an “E.” A diamond indicates further evidence is needed to support the claim.

Complex systems require complex assurance cases and they may be difficult for novice users to understand and build. One way to make assurance cases easier to share and use across domains has been to abstract similarities for reuse via patterns [53]. Patterns are meta-models of common argument structures and can be described and catalogued for retrieval and reuse. While useful for an expert in building assurance cases, these patterns may not be usable by a novice who may have too many degrees of freedom to concretize an abstraction.

This chapter proposes providing better guidance through a different level of abstraction, a template-like model called an “assurance recipe.” The user is provided a structure and pattern for an assurance case and is guided to select “ingredients” for a set of options based on domain knowledge and expertise. The recipes can be further customized for specific domains and then parameterized for easy user instantiation, such as the non-traditional domain of SB where users will not normally be familiar with building assurance cases.

The first section is a pre-study to understand whether the requirement for commonality exists in SB. The subjects of the pre-study are teams who earned a gold medal in the iGEM competition between 2015 and 2017 to understand whether there is a similar set of common approaches typically used for safety. Information and data manually extracted from those subjects was used in development of common recipes. Finally, this chapter presents a feasibility study to demonstrate how end-users could apply the recipes in practice. The recipes proposed are based on the emerging safety-critical discipline of SB [188], but the concepts could also be useful for other safety-critical disciplines.

3.1 Background and Related Work

Researchers have recognized a need for modularity and reuse of patterns which can facilitate design and understanding of assurance cases. To address the difficulty of assuring and certifying electronics systems in the aerospace industry, Ruiz et al. [183] suggest combining Case-Based Reasoning (CBR) as a way to represent, retrieve, and reuse previously assured safety cases. CBR analysis is based on the theory that similar problems have similar solutions. Well-developed patterns should help develop best practices because it will be easier to share domain knowledge and rationales. Indeed, “experts in technical fields such as medicine, engineering, planning, and finance rely on past cases to generate hypotheses about new situations.” [183]

For general safety-critical domains Conmy and Bate [45] propose a method to understand reuse of software components in different contexts, such as when a software module needs to be verified on new hardware. A key insight is that evidence used to verify a software module in one context is not necessarily sufficient or appropriate in another. They combine Component-Based Software Engineering (CBSE) concepts with argument fragments to guide the user.

Hawkins and Kelly [92] present the concept of a catalog of argument patterns to describe claims that could apply to any software assurance case. Similar to software design, software argument patterns are abstractions of common strategies or best practices. This concept was expanded by Szczygielska and Jarzebowicz [200] who proposed an online repository for assurance case patterns with a focus on universal application and uniformity across different industry domains for high-integrity systems.

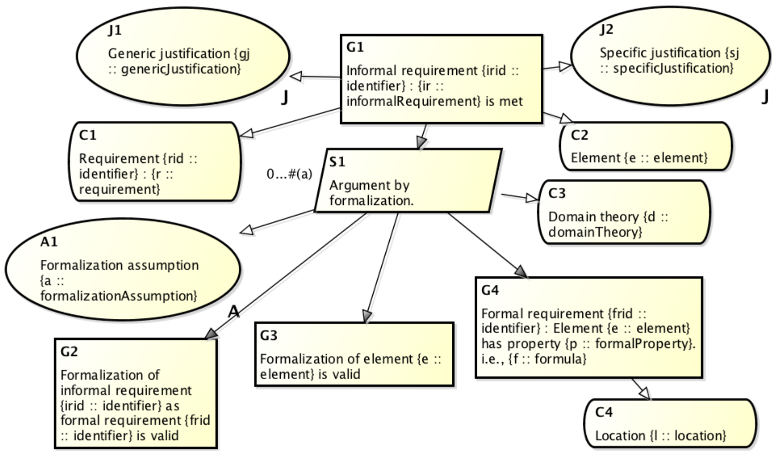

J1
Generic justification {gj :: genericJustification}
J
G1
Informal requirement {irid :: identifier} : {ir :: informalRequirement} is met
J2
Specific justification {sj :: specificJustification}
J
C1
Requirement {rid :: identifier} : {r :: requirement}
0...#(a)
S1
Argument by formalization.
C2
Element {e :: element}
C3
Domain theory {d :: domainTheory}
A1
Formalization assumption {a :: formalizationAssumption}
A
G4
Formal requirement {frid :: identifier} : Element {e :: element} has property {p :: formalProperty}. i.e., {f :: formula}
G3
Formalization of element {e :: element} is valid
G2
Formalization of informal requirement {irid :: identifier} as formal requirement {frid :: identifier} is valid
C4
Location {l :: location}

Figure 3.2: A high-level pattern based on Denney and Pai.

The recipes in this work are influenced heavily by the work of Denney and Pai who developed a foundation for a theory of assurance patterns, or a pattern for patterns [53]. Their work formalizes definitions of argument structures and argument patterns, and provides an algorithm for instantiating their patterns. The recipes in this chapter are based on their Claim Formalization Pattern shown in Figure 3.2.

3.2 Pre-study

For the pre-study, this chapter examined three years of projects from the iGEM competition, which is a yearly competition for teams of students from high school through to graduate school. This competition has grown rapidly in popularity, with 314 teams from over 40 countries in 2017 [102]. The iGEM competition provides a plentiful source of real-world projects because teams are encouraged to share information in an open-source database. Teams must also explicitly discuss safety of their projects, whereas private industry or academic institutions might choose to protect their designs through confidentiality or intellectual property laws.

Teams can attempt to earn a bronze, silver, or gold medal for satisfying increasingly stringent criteria. Gold-medal teams must accomplish Integrated Human Practices, which asks teams to “consider whether their projects are safe, responsible, and good for the world.” [255] The pre-study is limited to gold-medal teams from 2015-2017 based on the assumption they addressed safety to earn the gold medal.

The most common cellular chassis for iGEM projects is the K-12 strain of *E. coli*, a bacterium which does not inherently produce toxins and can be handled in a Safety Level One lab. By far, most experiments are performed in a Safety Level One lab using organisms which are generally regarded as safe (GRAS) [278]. However, even the relatively safe K-12 strain of *E. coli* could pose safety-critical risks outside the lab, because SEBOs have added functionality and behavior and can evolve or adapt to environmental stress. Also, iGEM teams are not required to perform the extensive testing needed to ensure their safety. Indeed, “it would be difficult for any team to gain real-world biosafety approval for their applications (*e.g.*, for medical use or environmental release).” [90]

3.2.1 Categorization Methodology

The safety pages of gold-medal iGEM teams from 2015-2017 provided 334 projects which fell into the following categories of safety features.[1]

- Containment (Con): the SEBOs and their products are safely contained in the lab and are not intended for release into the environment.
- Kill-switch (KS): upon unwanted evolution or mutation, a process is triggered to actively kill the SEBOs, often through lysis of cell membranes.
- Auxotrophy (Aux): the SEBOs cannot survive without the presence of a specific chemical or food source.
- Degradation (Deg): the SEBOs or their products degrade naturally over time when exposed to certain environmental conditions.
- Sterility (Ster): the SEBOs’ offspring are engineered to be sterile.
- GRAS: the organisms and their products are in Risk Group 1 as defined by the FDA or the NIH.

Some of the categorization required making subjective determinations about which safety features were used. Also, some teams used terminology inconsistently, such as the 2016 Wageningen Team which described auxotrophy as a kill-switch [275]. These subjective decisions represent a threat to the validity

[1] A spreadsheet of the teams is available at https://sites.google.com/view/splc-dnafeatures/emse, under the Journal Version (EMSE), RQ2.

of the categories and statistics, but nevertheless should be a relatively fair representation of the 334 projects.

Year	# Gold Medals	KS	Con	GRAS	Aux	Deg	Ster
2015	114	7.02%	14.04%	50.00%	3.51%	1.75%	0.88%
2016	111	12.61%	20.72%	43.24%	2.70%	5.41%	0.00%
2017	109	13.76%	16.51%	44.04%	6.42%	0.00%	0.00%

Table 3.1: Top safety features from recent iGEM competitions.

Year	Gold Medals Awarded	Auxotrophy	Degradation	Sterility
2015	114	3.51%	1.75%	0.88%
2016	111	2.70%	5.41%	0.00%
2017	109	6.42%	0.00%	0.00%

Table 3.2: Less-common safety features from recent iGEM competitions.

In fact, some teams acknowledged on their safety pages that even though they used organisms or materials which were GRAS, there nevertheless was some unknown level of risk associated with their projects. One team stated: “If the experiment did not proceed carefully that result [sic] in the leak of the culture medium, the bacteria may breed at a considerable speed especially for the human cell culture medium and serum we would use in our lab. Thus it will contaminate the lab.” [271]

Other examples of teams expressing doubts or concerns about safety include:

- “[A]ny improper disposal of ccdB protein and plasmids containing ccdB gene will pose serious results to the natural bacterial communities . . . ” [273]

- “Since the toxic proteins we choose are relative [sic] new, we cannot find their degradation period, degradation condition and more information related to safety. Thus we design a series of future experiments.” [276]

- “If you ask us whether our project is safe or not, we will say yes and it won’t do any harm to humans and the environment.” [251]
- “[I]ntegration of this modified probiotic strain into patients’ gut floras however would require a whole lot of other biosafety considerations . . . ” [260]
- “When we consider this in the aspect of evolution, we cannot predict the possibility that there’s an unknown toxin can be produced by PCC6803 . . . ” [270]
- “The pathway responsible for breaking down naphthalene was extracted from the Pseudomonas putida PpG7. This organism is safety class 1, but since there have been some cases where this bacteria have infected humans we took extra safety precautions when working with it; it was handled and kept in a fume hood.” [274]
- “There is a non-zero probability of our strains mutating to a pathogenic serotype.” [277]

Most teams base the safety of their projects on the mere fact their organisms are GRAS, but some teams add safety features into their projects. Table 3.1 shows usage of safety features. The percentages do not add up to 100% for a few reasons. First, many teams did not complete their webpage stubs for the Safety category, or it was otherwise unclear what their approach for safety was. It is likely those teams were relying on GRAS Only as their safety feature, meaning the percentages from Table 3.1 for GRAS Only are likely under-reported. Second, some of the teams considered more than one safety feature, such as the 2017 IONIS-PARIS team’s “four walls” design [256]. Finally, some teams entered projects which did not involve SEBOs directly, but instead were in silico contributions, such as developing or improving software or hardware.

3.3 Assurance Recipes

The concept of an assurance recipe begins with a pattern, specifically the Claim Formalization Pattern from Denney and Pai [53]. This section provides recipes for the two most-common safety features in Table 3.1. The first, the Containment Recipe, assumes SEBOs are not intended for release into the environment. The second, the Safety Mechanism Recipe, assumes SEBOs will have applications outside of the laboratory or production facility.

3.3.1 Containment Recipe

This recipe places focus on safety levels and associated risks, and is shown in Figure 3.3(a). It is intended to address the four different lab safety levels and the risks associated with organisms requiring those safety levels. The assumption is that a competent government agency has accurately declared the organisms to be from a specific risk group. The evidence needed will mostly be documentation of adequate training, along with proof of adequate logical and physical security measures needed to certify the lab. Table 3.3 suggests ingredients for the recipe from which the user can select the appropriate choices.

Variable	Ingredients
cf :: containmentFacility	1. Safety Level One Lab 2. Safety Level Two Lab 3. Safety Level Three Lab 4. Safety Level Four Lab
rg :: riskGroup	1. Risk Group One 2. Risk Group Two 3. Risk Group Three 4. Risk Group Four
ht :: hazardType	1. cannot cause disease in healthy adults; 2. can cause treatable or preventable disease in humans; 3. can cause serious disease in humans which might not have a treatment or vaccine; 4. can cause serious disease in humans which has no known treatment or vaccine

Table 3.3: Suggested ingredients for the Containment Recipe.

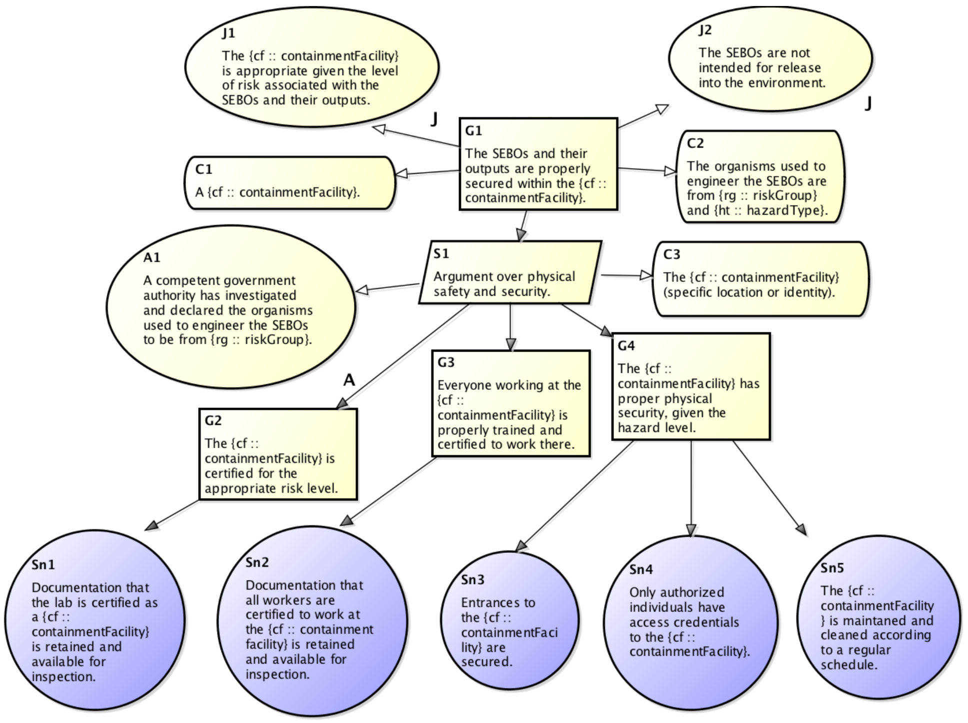

J1
The {cf :: containmentFacility} is appropriate given the level of risk associated with the SEBOs and their outputs.
J
J2
The SEBOs are not intended for release into the environment.
J
G1
The SEBOs and their outputs are properly secured within the {cf :: containmentFacility}.
C1
A {cf :: containmentFacility}.
C2
The organisms used to engineer the SEBOs are from {rg :: riskGroup} and {ht :: hazardType}.
A1
A competent government authority has investigated and declared the organisms used to engineer the SEBOs to be from {rg :: riskGroup}.
A
S1
Argument over physical safety and security.
C3
The {cf :: containmentFacility} (specific location or identity).
G2
The {cf :: containmentFacility} is certified for the appropriate risk level.
G3
Everyone working at the {cf :: containmentFacility} is properly trained and certified to work there.
G4
The {cf :: containmentFacility} has proper physical security, given the hazard level.
Sn1
Documentation that the lab is certified as a {cf :: containmentFacility} is retained and available for inspection.
Sn2
Documentation that all workers are certified to work at the {cf :: containment facility} is retained and available for inspection.
Sn3
Entrances to the {cf :: containmentFacility} are secured.
Sn4
Only authorized individuals have access credentials to the {cf :: containmentFacility}.
Sn5
The {cf :: containmentFacility} is maintaned and cleaned according to a regular schedule.

Figure 3.3: The Containment Recipe.

3.3.2 Safety Mechanism Recipe

The recipe in Figure 3.4(b) addresses the most-common SEBO safety mechanisms. The assumption is that the SEBOs will be released into the environment and that they pose some risk of harm. The evidence will heavily rely on wet-lab experimental data. The safety and security of the SEBOs are included as sub-goals, but there may not be sufficient experimental evidence to support them. A node which does not have sufficient evidence could be annotated to indicate which laboratory experiments are needed to support the sub-goals. Table 3.4 suggests ingredients for the Containment Recipe.

3.4 Feasibility

This section assesses whether the recipes can be instantiated based on four iGEM projects. The first is the 2017 University of Nebraska - Lincoln (UNL) team.

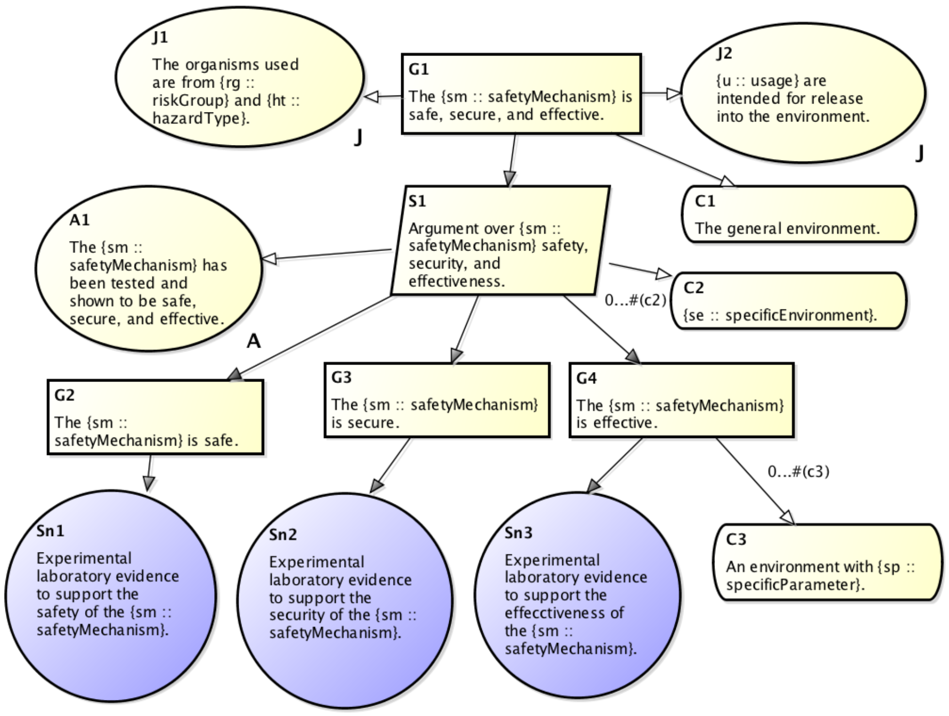

J1
The organisms used are from {rg :: riskGroup} and {ht :: hazardType}.
J
G1
The {sm :: safetyMechanism} is safe, secure, and effective.
J2
{u :: usage} are intended for release into the environment.
J
S1
Argument over {sm :: safetyMechanism} safety, security, and effectiveness.
A1
The {sm :: safetyMechanism} has been tested and shown to be safe, secure, and effective.
A
C1
The general environment.
C2
{se :: specificEnvironment}.
0...#(c2)
G2
The {sm :: safetyMechanism} is safe.
G3
The {sm :: safetyMechanism} is secure.
G4
The {sm :: safetyMechanism} is effective.
0...#(c3)
Sn1
Experimental laboratory evidence to support the safety of the {sm :: safetyMechanism}.
Sn2
Experimental laboratory evidence to support the security of the {sm :: safetyMechanism}.
Sn3
Experimental laboratory evidence to support the effecctiveness of the {sm :: safetyMechanism}.
C3
An environment with {sp :: specificParameter}.

Figure 3.4: The Safety Mechanism Recipe.

Variable	Ingredients
sm :: safetyMechanism	1. Kill-switch 2. Auxotrophy 3. Degradation 4. Sterility
u :: usage	1. The SEBOs 2. Only the SEBOs' outputs
se :: specificEnvironment	1. Soil 2. Water table 3. [Specific species habitat] 4. Atmosphere 5. Rivers 6. Freshwater rivers or lakes 7. Saltwater lakes or oceans 8. Human body 9. [Non-human] body
sp :: specificParameter	1. [Temperature Range] 2. [pH Range] 3. [Aerobic/Anaerobic] environment 4. [Natural/Specific frequencies/Absence] of light 5. [Presence/Absence] of nutrients 6. [Altitude range]

Table 3.4: Suggested ingredients for the Safety Mechanism Recipe.

They thoroughly documented the safety training each member had to achieve before working in the lab, which provided a logical source for instantiation of the Containment Recipe. The UNL team listed themselves as working in a Safety Level 1 lab, with organisms from Risk Group 1, and were thus the first hazardType. For evidence, the students needed to demonstrate they followed protocols for proper containment. Their documentation would be appropriate to fill in evidence nodes such as Sn2. They wrote: “We took a total of 6 safety modules including a Biosafety Level 1 course before we were allowed to work in the lab. All modules that required a quiz to be taken had to be completed with 80% proficiency.” [262]. The assurance case instantiation is shown in Figure 3.5.

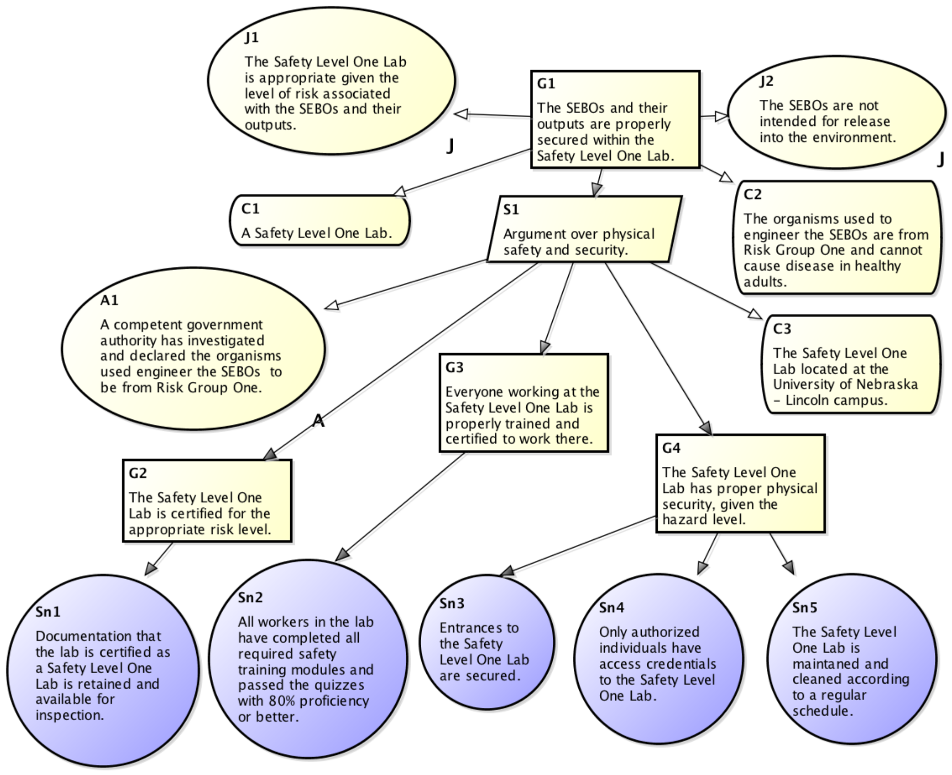

J1
The Safety Level One Lab is appropriate given the level of risk associated with the SEBOs and their outputs.
J
G1
The SEBOs and their outputs are properly secured within the Safety Level One Lab.
J2
The SEBOs are not intended for release into the environment.
J
C1
A Safety Level One Lab.
S1
Argument over physical safety and security.
C2
The organisms used to engineer the SEBOs are from Risk Group One and cannot cause disease in healthy adults.
A1
A competent government authority has investigated and declared the organisms used engineer the SEBOs to be from Risk Group One.
A
C3
The Safety Level One Lab located at the University of Nebraska – Lincoln campus.
G3
Everyone working at the Safety Level One Lab is properly trained and certified to work there.
G2
The Safety Level One Lab is certified for the appropriate risk level.
G4
The Safety Level One Lab has proper physical security, given the hazard level.
Sn1
Documentation that the lab is certified as a Safety Level One Lab is retained and available for inspection.
Sn2
All workers in the lab have completed all required safety training modules and passed the quizzes with 80% proficiency or better.
Sn3
Entrances to the Safety Level One Lab are secured.
Sn4
Only authorized individuals have access credentials to the Safety Level One Lab.
Sn5
The Safety Level One Lab is maintaned and cleaned according to a regular schedule.

Figure 3.5: Assurance case fragment instantiation for the Containment Recipe.

Kill switches are the most-common approaches to non-containment safety, and Figure 3.6 shows an instantiation of the Safety Mechanism Recipe based on the Hok/Sok kill switch implemented by a University of Maryland team [261]. The key feature is the evidence based on wet-lab experiments showing that the kill switch will trigger within 30 seconds of mutation.

The next-most common safety feature is to engineer auxotrophy into SEBOs.

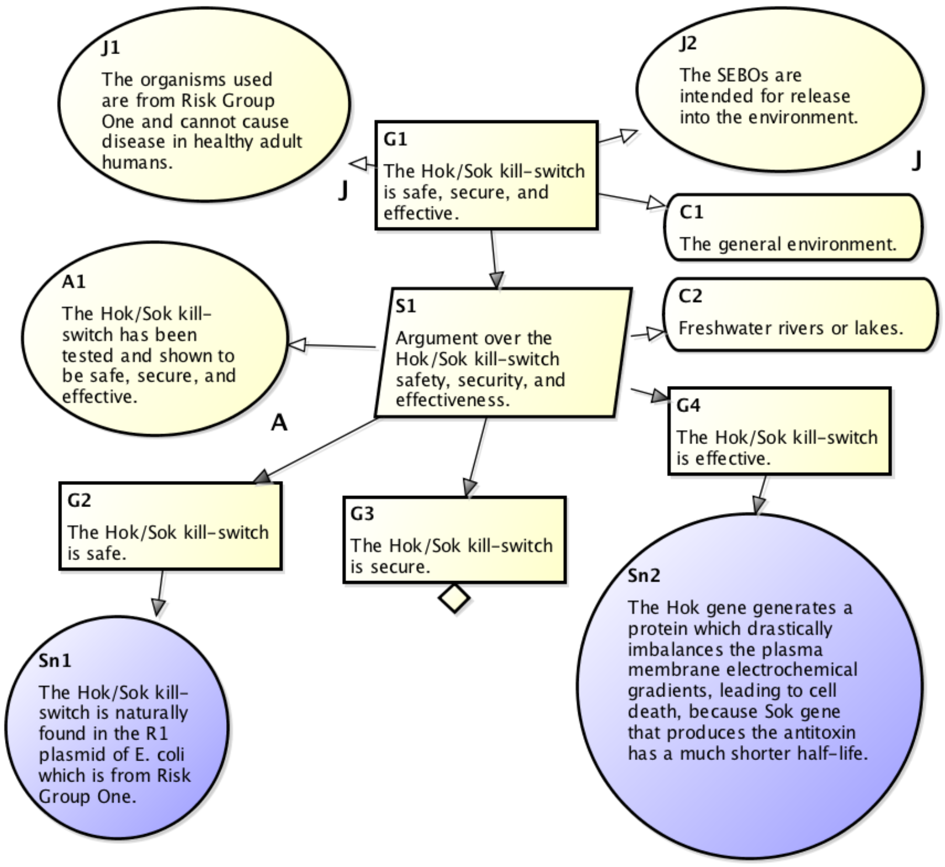

J1
The organisms used are from Risk Group One and cannot cause disease in healthy adult humans.
J
G1
The Hok/Sok kill-switch is safe, secure, and effective.
J2
The SEBOs are intended for release into the environment.
J
C1
The general environment.
A1
The Hok/Sok kill-switch has been tested and shown to be safe, secure, and effective.
A
S1
Argument over the Hok/Sok kill-switch safety, security, and effectiveness.
C2
Freshwater rivers or lakes.
G4
The Hok/Sok kill-switch is effective.
G2
The Hok/Sok kill-switch is safe.
G3
The Hok/Sok kill-switch is secure.
Sn1
The Hok/Sok kill-switch is naturally found in the R1 plasmid of E. coli which is from Risk Group One.
Sn2
The Hok gene generates a protein which drastically imbalances the plasma membrane electrochemical gradients, leading to cell death, because Sok gene that produces the antitoxin has a much shorter half-life.

Figure 3.6: Assurance case fragment instantiation for the Safety Mechanism Recipe for a kill switch.

Figure 3.7 shows an assurance case fragment based on the 2016 Wageningen team [275] which implemented an auxotrophic system based on prior work [149]. The team designed SEBOs to produce a chemical which helps bees defend against mites. The SEBOs were engineered to require a synthetic amino acid called BipA to survive, a compound which beekeepers could add into sugar water for feeding a bee colony. If the SEBOs were to escape the beehive, they would die within 72 hours without BipA.

Last is an assurance case for degradation based on the 2016 Formosa team which developed a pesticide called Pantide [254]. Because it was a novel toxin, the team performed experiments to determine that it sufficiently degrades within two hours of exposure to natural light at 36.8° C. This is shown in Figure 3.8.

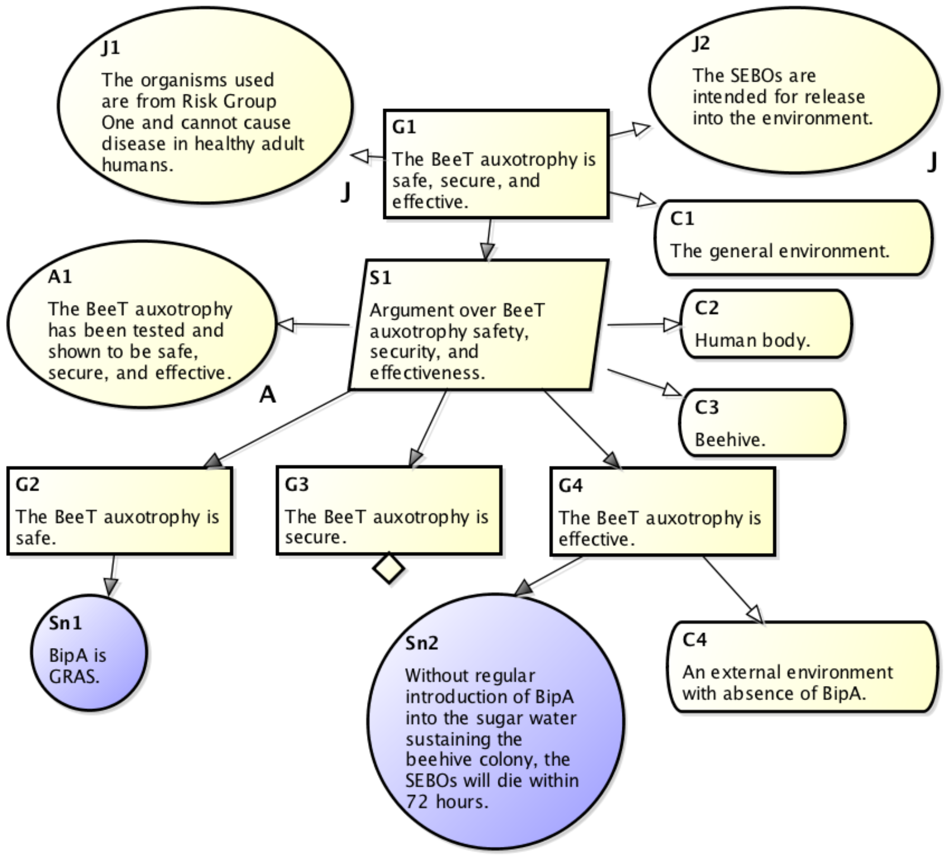

J1
The organisms used are from Risk Group One and cannot cause disease in healthy adult humans.
J
G1
The BeeT auxotrophy is safe, secure, and effective.
J2
The SEBOs are intended for release into the environment.
J
C1
The general environment.
A1
The BeeT auxotrophy has been tested and shown to be safe, secure, and effective.
A
S1
Argument over BeeT auxotrophy safety, security, and effectiveness.
C2
Human body.
C3
Beehive.
G2
The BeeT auxotrophy is safe.
G3
The BeeT auxotrophy is secure.
G4
The BeeT auxotrophy is effective.
Sn1
BipA is GRAS.
Sn2
Without regular introduction of BipA into the sugar water sustaining the beehive colony, the SEBOs will die within 72 hours.
C4
An external environment with absence of BipA.

Figure 3.7: Assurance case fragment instantiation for the Safety Mechanism Recipe for auxotrophy.

3.5 Conclusion

This chapter presented the idea of an assurance recipe and demonstrated how it can be applied in the non-traditional domain of synthetic biology. Although the assurance recipes are based on the most-common safety features of SEBOs from iGEM projects, they are intended for general application to any SEBOs, and should be helpful for other safety-critical disciplines. Assurance recipes and ingredients can facilitate use and reuse by domain experts who lack expertise with building assurance cases.

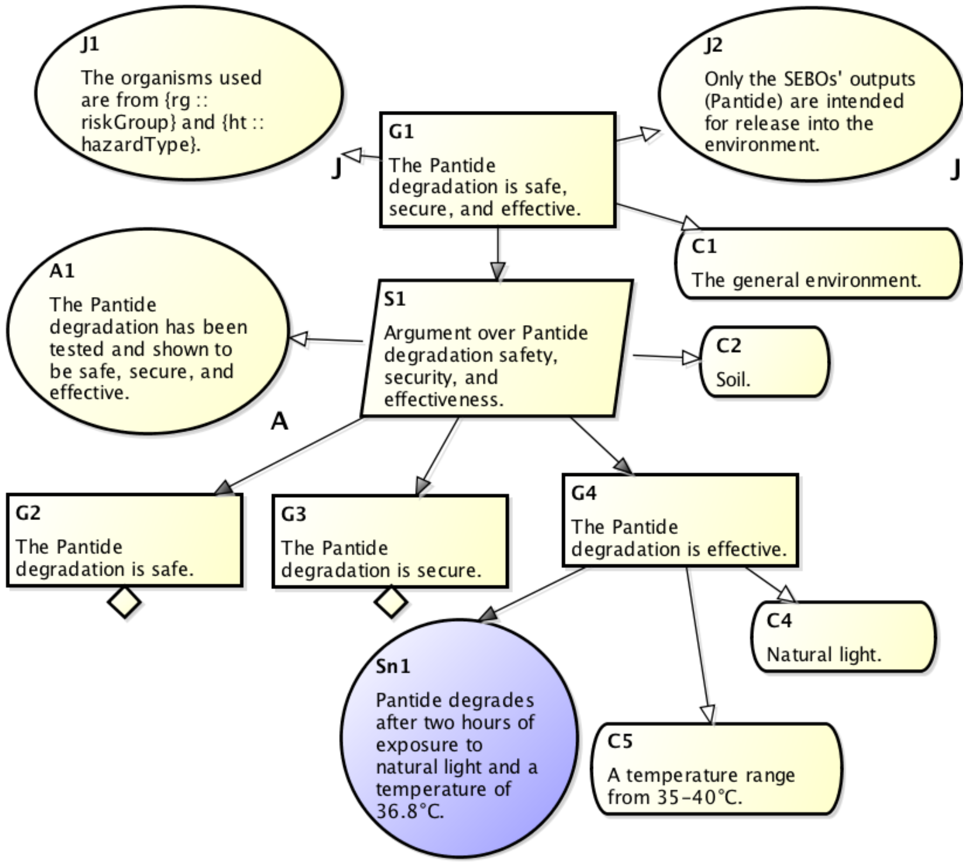

J1
The organisms used are from {rg :: riskGroup} and {ht :: hazardType}.
J
G1
The Pantide degradation is safe, secure, and effective.
J2
Only the SEBOs' outputs (Pantide) are intended for release into the environment.
J
C1
The general environment.
A1
The Pantide degradation has been tested and shown to be safe, secure, and effective.
A
S1
Argument over Pantide degradation safety, security, and effectiveness.
C2
Soil.
G2
The Pantide degradation is safe.
G3
The Pantide degradation is secure.
G4
The Pantide degradation is effective.
C4
Natural light.
Sn1
Pantide degrades after two hours of exposure to natural light and a temperature of 36.8°C.
C5
A temperature range from 35-40°C.

Figure 3.8: Assurance case fragment instantiation for the Safety Mechanism Recipe for degradation.

Chapter 4

Building Feature Models from Synthetic Biology BioBricks.

The most basic design in SB is the transcription unit shown in Figure 2.1. A transcription unit is comprised, in sequence, of four basic DNA parts: a promoter, an RBS, a protein coding sequence, and a terminator. An engineer browsing the iGEM registry would find 3,912 promoters [263], 747 RBSs [264], 9,699 protein sequences [252], and 503 terminators [272] from which to choose. That means there are $3,912 \times 747 \times 9,699 \times 503$ possible combinations, which is over 14 trillion, for the most basic design in SB. This is ultimately a very low boundary for the actual number of possible combinations, because a transcription unit can be 5,000bp or more, which is $4^{5,000}$ possible combinations, a number far larger than the number of estimated hydrogen atoms in the universe (10^{80}) [194].

The DNA parts are often described as LEGO® pieces that can be combined in many ways to form new genetic devices. However, these LEGO® pieces come with no building or assembly instructions, and often lack important information for reusability. An engineer might begin a project by developing a blueprint for their intended SEBO and behavior, and plan the type of features their system should have. An example of a feature would be a part that produces a fluorescent protein, visible to the naked eye, in the presence of a particular chemical. The engineer would need to find the corresponding parts in a repository or research literature. However, a specific architecture is required to ensure the various features work together, and this can require significant domain knowledge and expertise.

Similar to the BioBrick repository for programming SEBOs with modular and reusable parts, current software development practices rely heavily on code modularity and reusability. Programmers create open-source projects emphasizing plug-and-play interfaces for implementing code in a wide variety of applications. Open-source repositories such as GitHub have emerged where developers can reuse, and even help improve, code libraries. Well written libraries and components should reflect low coupling and high cohesion, a core tenet of software engineering [97].

However, even the best code libraries can become extremely large and difficult to maintain. As new packages are added to existing projects, it is not easy to determine whether prior dependencies and constraints are still valid, or whether new ones were added. To address these issues, software product line (SPL) engineering has become a best practice for modeling and managing families of software systems. The SPL community has turned to open-source systems and embraced concepts of commonality and variability to define open-source product lines [195]. Montalvillo and Díaz have proposed techniques to aid SPL practices within GitHub [154].

Because large open-source code bases and a large DNA parts database present similar challenges for engineers hoping to use them in their designs, this chapter explores whether the SPL practice of feature modeling can be applied to SB and the open-source database of DNA BioBricks.

4.1 Background

The process of engineering SEBOs starts with a model which is then implemented into strands of DNA to be inserted into a living cell. The organism is like a compiler: it treats the inserted DNA code as its own native DNA, and then translates it to machine-level code which is transcribed into proteins which perform or enable different functions. While machine code for computers is written in 1s and 0s, the machine code for biology is written in the four DNA bases of adenine (A), thymine (T), cytosine (C), and guanine (G).

This analogy of programming biology is not a new concept. There is a programming language called the Synthetic Biology Open Language (SBOL) which defines a common way to represent biological designs [280]. Researchers have used SB to enable: 1) a context-free grammar using BioBricks [31]; 2) automated design of genetic circuits with NOT/NOR gates [164]; and 3) bacterial networks to use DNA for data storage [23, 201]. Other SB organic programs have achieved traditional computer science constructs such as: 1) a genetic oscillator [61]; 2) a genetic toggle switch [83]; and 3) a time-delay circuit [222].

The discipline of SB breaks down biological processes into smaller functions, each of which can be represented by a DNA part. These parts can be assembled in various combinations to make new functions. The largest open-source repository of DNA parts is called the Registry of Standard Parts [265] which contains over 45,000 entries. Teams from the iGEM competition submit their parts to the registry with various information about their usage and efficacy. Although iGEM teams are the most frequently documented users of the repository, anyone can search it to find appropriate parts for their designs.

iGEM describes itself as a competition where teams “design, build, test, and measure a system of their own design using interchangeable biological parts and standard molecular biology techniques.” Each year about 6,000 students participate, designing projects which often address local problems such as pollution mitigation. Teams are judged by community experts and can be awarded a medal, either bronze, silver, or gold, depending on the impact and contributions of their project. Gold-medal teams must achieve several goals such as modeling their project, demonstrating their work through experimentation, collaborating with other teams, addressing safety concerns, improving pre-existing parts or projects, and contributing new parts.

4.1.1 Motivating Example

The following motivating example is derived from a case study demonstrating the potential of SPL engineering for SB. A kill switch is a safety mechanism which triggers cellular death, typically by engineering cells to produce proteins which destroy cellular membranes. Common triggers include exposure to specific chemicals, temperature ranges, pH levels, or frequencies of light. The feature model for the kill switch in Figure 4.1 was developed by manually reviewing all of the wiki pages from the 110 teams who earned a gold medal in the 2017 iGEM competition, and is the same set used for prior work on assurance cases [70]. Of those 110 teams, 14 mentioned some type of kill switch in their design. Thus, a complete model for all possible kill switches would be much larger, and Figure 4.1 only represents a small subset of the total configuration space.

If a synthetic biologist were to do a free-text search for “kill switch” (without the quote marks in the search box) at the registry website, the query produces 145 results [257]. A similar search for “kill-switch” (again without quote marks) produces 423 hits [258]. The page on parts by type for “cell death” features 51 parts [253], and the Biosafety page has a list of 25 kill-switch parts [266]. It is not obvious which search produces the most relevant results, and it would require manual inspection of each result to determine whether the parts were suitable for the intended design.

What if, instead of starting from scratch, the synthetic biologist begins this process with the feature model in Figure 4.1? From this model, users could quickly see the architecture of their system. They would understand that a kill

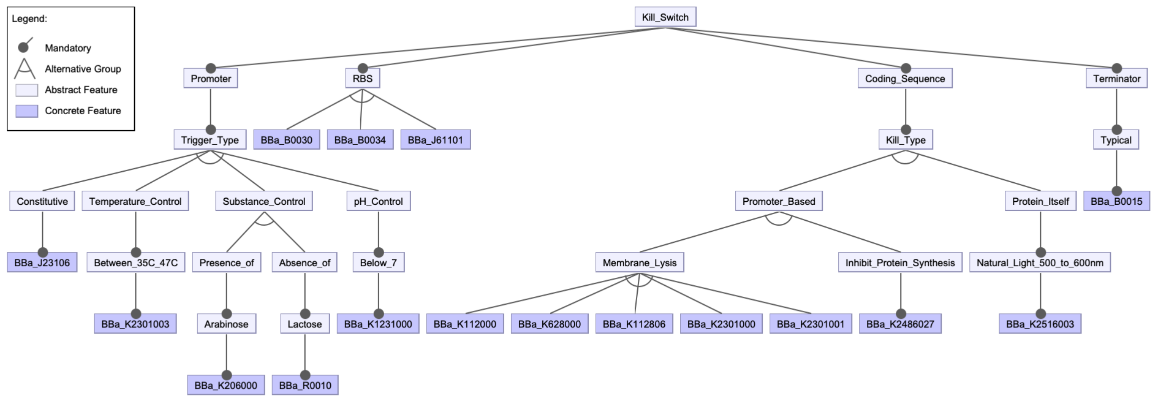


Figure 4.1: A feature model for a kill switch.

switch is comprised of four basic parts: a promoter, an RBS, a coding sequence, and a terminator. Instead of having to look at hundreds of possible parts, and not knowing exactly what to type in a free-text search box, users could see that there are two important points of variability for a kill switch: the trigger type and the kill type.

If users wanted to test the effectiveness of different types of kill switches in a specific system, they could slice this model to get a specific set of products. Moreover, slicing the model would become more important as more features are added. Users could also design experiments to test products which have not yet been implemented or analyzed in a laboratory. Upon completion of a successful experiment, users could add data and notes as annotations.

This type of information is not readily available within the iGEM parts registry. If teams successfully implement a kill switch, they may or may not share their results as free text on their webpages with specific information about the parts they used, and teams do not provide consistent levels of documentation or specificity. There is no central source for this information and locating it would require several manual searches.

4.2 Related Work

This study follows a long line of research on software product lines, and does not attempt to summarize all of that work. There are several good surveys on this topic [18, 205]. Product-line engineering has been applied in many emerging domains recently including drones and nanodevices [42, 145]. There is also a push towards open-source product lines [142, 154, 195] and the concept of a software ecosystem, where the community modifies and customizes product lines using a common platform and look [173].

The most similar work is that of Lutz et al. [145] who studied a family of DNA nanodevices. While they also looked at DNA, they studied chemical reaction networks (CRNs), rather than a living synthetically engineered organism. Their line of organic programming leveraged CRNs, which are sets of concurrent equations that can be compiled into single strands of DNA [229]. CRNs have been shown to be Turing complete and are not part of living organisms because they do not require transcription and translation of DNA code [64, 67].

Other researchers have examined relationships within the BioBrick repository from a network perspective to gain an understanding of its complexity, and they also consider it to be a software ecosystem [214]. This case study is different from prior work by showing how domain engineering can build a family of synthetic biology products to be analyzed and reasoned about using traditional SPL techniques.

4.3 Organic Software Product Lines

The SPL community has asked whether open-source applications such as the Linux Kernel should be considered product lines given that they were not managed and developed in the traditional manner [142, 195]. The consensus has been that these applications should be considered product lines, and the community now has a broader view of what constitutes an SPL. Turning to SB and organic programs, it seems logical to ask whether there could be a mapping between traditional SPL engineering to enable managed development and reuse of DNA parts.

Clements and Northrop stated that the output of domain engineering should contain: 1) a product-line scope; 2) a set of core assets; and 3) a production plan [43]. In application engineering the production plan and its scope help build and test individual products. The kill-switch model focuses primarily on domain engineering, but there are tangential concepts related to application engineering through the context of organic programs.

4.3.1 Assets

Assets in traditional product lines can include a software architecture, reusable software components, performance models, test plans and test cases, and other design documents. Organic programs have similar elements. One is usage of SBOL to define the functionality of assembled DNA parts, and this serves as an important design document for individual features. The SBOL models can be composed and aggregated into larger composite models, and should include the specific DNA sequences required for each part, as well as a visual representation (e.g., Figure 2.1).

The DNA sequence itself is the reusable software component. Like code it is not tangible, but must be implemented as a program and compiled down to a machine level representation. DNA can be synthesized into a physical strand to be inserted into a compiler (the living organism) for translation to machine level code (via the biological process of transcription of DNA via RNA into proteins). Other assets such as test cases and test plans can be constructed which define either laboratory experiments or virtual simulations. Both help users evaluate a program's expected, versus observed, functionality. Additional assets in the form of design documents and documentation can be provided such as safety cases [70] and system-level architectures expressed through tools such as GenoCAD [31].

4.3.2 Domain Engineering

During domain engineering, users define the product-line scope by choosing a family of behavior such as a type of molecular communication or computation. When inserted into their host organisms at specific binding sites, the sets of DNA sequences that perform desired functionality define unique sets of products which exhibit commonality and variability. Finally, a production plan can be created in combination with a feature model and constraints. The feature model and constraints show how the DNA parts, as features, form a family of products, and annotations can include experimental notes on expected environmental conditions or other assumptions that are required for the program to run correctly.

4.3.3 Application Engineering

Application engineering involves combining the expected parts using standard DNA cloning techniques for insertion into a living organism [177]. As with traditional product lines, the engineer must comply with any constraints and compose only those products which are validly defined in the feature model, otherwise unexpected behavior may occur. As in traditional software, some constraints may be hard-coded into the program, while some may represent a domain expectation instead.

Thüm, et al. [205] in their survey of product lines discuss different implementations of product lines and analyses that can be performed at both the family and variability level. Organic product lines are another type of implementation and we can utilize common analyses and techniques from SPL engineering in this domain.

4.3.4 Case Study

Next is a case study to evaluate the feasibility of using product-line engineering to help synthetic biologists. The case study evaluates two research questions:

- RQ1: Does a DNA repository have functions with the characteristics of a software product line?
- RQ2: If the answer to RQ1 is yes, then is it possible to build feature models representing families of products from an existing DNA repository? If the repository has sets of products with software product line characteristics, it should be possible to use them to make a feature model.

The subject repository is the Registry of Standard Biological Parts as described in Section 7.2. The registry represents the largest open-source DNA repository with 47,934 parts and continues to grow by an average 2,995 parts each year. The parts are sorted into ten common biological functions: biosafety, biosynthesis, cell-to-cell signaling and quorum sensing, cell death, coliroid, conjugation, motility and chemotaxis, odor production and sensing, DNA recombinations, and viral vectors. This study focuses on biosafety, a subset of which includes kill switches. Based on these functions and characteristics, RQ1 is answered in the affirmative.

The feature model shown in Figure 4.1 represents manual review of all 2017 iGEM teams who earned a gold medal. The “Trigger” section will likely be the most interesting for synthetic biologists because they will want to engineer their kill switches to activate only under certain conditions. There are many other possible designs for kill switches which are not represented in this model, but it serves as a starting point for further discussion. Even this small model represents 882 valid products.

Kill switches in the model can be triggered under several different conditions: 1) temperature ranges; 2) presence or absence of specific chemicals; 3) low pH levels; or 4) exposure to specific frequencies of natural light. Each of those trigger conditions ends in leaf nodes which are the BioBrick IDs correlating to actual DNA sequences. The promoters, RBSs, coding sequences, and terminators show which BioBricks could be used to complete a transcription unit for a kill switch. The "Visualization" branch is optional and provides visual evidence to the naked eye that the kill switch is working through production of fluorescent proteins. The feature model shown in Figure 4.1 properly captures the characteristics of the BioBricks associated with kill switches, thus RQ2 is answered in the affirmative.

4.4 Conclusion

Feature models such as the one shown in Figure 4.1 can help synthetic biologists guide the design-build-test process by making it easier to more quickly find appropriate parts for their SEBOs. This shows that feature models, a traditional software-engineering concept, can also be applied to a non-traditional domain such as SB.

Chapter 5

The Need for Soft Law to Regulate Synthetic Biology[1]

This chapter examines some of the regulatory issues regarding SB, with particular concern that despite several international collaborative efforts 10-15 years ago, there currently are few formal legal requirements with respect to SB. Part of the problem may be that SB is a challenging, interdisciplinary field, and regulators do not have requisite expertise to adequately and efficiently regulate an emerging and rapidly advancing technology. It is difficult to fine-tune legal requirements so that they do not stifle legitimate progress, yet in turn provide enough protection and transparency to satisfy the public. The dual-use nature of SB suggests the threats to life and the environment could be significant, but so far, there has been little more than discussion of potential threats.

In 1999, the U.S. Commission on National Security/21st Century issued its Phase I report to draw attention to the most likely trends or emerging threats, one of which was that: "[r]apid advances in information and biotechnologies will create new vulnerabilities for U.S. security." [91] Although this report did not use the phrase "synthetic biology" (SB), the report was concerned with the proliferation of weapons of mass destruction (WMDs), whether nuclear, chemical, or biological. If the report had issued ten years later, the authors would have probably specifically

[1]This chapter was written for *Jurimetrics*, the law journal of the Science & Technology Law section of the American Bar Association [69]. The writing style incorporates opinions and arguments which might otherwise be unexpected or inappropriate for scientific papers.

mentioned SB, a new technology with dual-uses which could help proliferate biological WMDs.

In fact, in 2010 (eleven years later), the Presidential Commission for the Study of Bioethical Issues directly assessed the dual-use nature of SB and recommended “that the field of synthetic biology can proceed responsibly by embracing a middle ground–an ongoing process of prudent vigilance that carefully monitors, identifies, and mitigates potential and realized harms over time. Responsible stewardship requires clarity, coordination, and accountability across the government. While new agencies, offices, or authorities are not necessary at this time, the Executive Office of the President should lead an interagency process to identify and clarify, if needed, existing oversight authorities and ensure that the government is informed on an ongoing basis about developments, risks, and opportunities as this field grows. This process must be undertaken by an office with sufficient authority to bring together all parts of the government with a stake in synthetic biology and be sufficiently authoritative to effectively engage or oversee engagement with foreign governments.” [174]

This recommendation implicitly suggests that in 2010 there was no need for new federal statutes, federal agencies, or any formally binding international agreements. Instead, the guidance was simply “prudent vigilance” and “responsible stewardship” within and among existing federal agencies which might have authority to regulate SB. If serious concerns, risks, or threats were to arise, stakeholders from all parts of the U.S. government would take the lead domestically and internationally. Given the state of the art of SB at the time, this approach seemed logical.

Researchers argued that SB was not as easy or accessible to the do-it-yourself community (DIYers) as some had warned, and the claim that SB parts were modular and easily composable like building blocks was inappropriate.

“The problem is that many parts have not been characterized well. They haven’t always been tested to show what they do, and even when they have, their performance can change with different cell types or under different laboratory conditions.” [127]

Also, it is somewhat misleading for SB proponents to analogize the modular DNA parts to LEGO® pieces, as if they can simply be snapped together in an easy plug-and-play design [127]. The reality is that many parts are not documented or tested well enough to readily incorporate them into new projects, especially if being used in different chassis organisms, and it is difficult to engineer novel cellular behavior that is truly orthogonal to its native DNA code [127]. Researchers have complained that it is often nearly impossible to reproduce anyone else’s SB experiments due to lack of full DNA sequence sharing or full descriptions of the genetic networks [170]. Lastly, transforming biology into an engineering discipline is not a straightforward task, because life is not static like a traditional mechanical device [190]. The goals of abstraction, modularity, and standardization have more fluidity when applied to living cells [190]. Notably, cells can mutate and evolve, and we do not have a complete understanding of the underlying biological systems being manipulated [190].

However, many of these drawbacks and challenges have been mitigated over that last ten years through better sharing of information and open-source databases of standardized parts using standardized formats, including more complete documentation to facilitate reproducibility of experiments [151]. There is active work by biologists and computer scientists to improve on a standardized language called Synthetic Biology Open Language (SBOL), which is now commonly used to describe basic DNA parts. SBOL allows research groups to more easily replicate experiments and share wet-lab results or distribute work from different stages of the design-build-test cycle to allow for specialization among various areas of expertise [80]. Standardization efforts such as SBOL allow for the reuse of previously validated designs, improve the ability to publish results, and help reduce ambiguity or vagueness in definitions. Today, SB has matured into an engineering discipline compared to the science of biology [80].

Consider that every year, high school and college students from around the world learn basic SB safety and skills for the iGEM competition held in Boston by MIT [102]. The competition has grown rapidly in popularity, with 314 teams from over 40 countries in 2017, including a team from Ghana which was able to make its own incubator using LEGO® pieces [103]. Emerging DNA synthesis companies allow researchers to order synthetic genes coded in uploaded text files and receive the genes “within a matter of days.” [56]

A decade of progress has produced several real-world SB successes, such as the synthesis of artemisinin, an anti-malarial drug which is difficult to produce naturally [168]. Medical research has improved through studies of microRNA strands (roughly 22bp) which are opening new avenues for SB to improve or create novel gene therapies for diseases which were otherwise considered untreatable [119]. The world’s ever-increasing demands for energy could be mitigated through industrial production of biofuels, something which was though possible back in 2008 [131], but now seems ever more likely [108].

However, each beneficial innovation has a theoretical dual-use counter. The ability to synthesize drugs is itself a specific dual-use technology, as it is now possible to synthesize opioids in yeast [79], which could result in synthesized heroin or fentanyl [59]. Gene therapies could open the door to the creation of specialized diseases targeting specific individuals [96]. The agricultural sector could face novel or more pathogenic strains of existing bacteria or viruses which could damage traditional biofuel sources such as ethanol and soybeans [225].

Despite last decade’s rapid advances and lowering of costs to perform DIY

SB experiments for academia, industry, and DIY enthusiasts, it appears “prudent vigilance” and “responsible stewardship” were more like “laissez-faire” in application. That is, from 2010-2018, there was little or no formal governmental response to SB dual-uses. Furthermore, it is inappropriate to consider SB dual-uses as a U.S.-only problem or inappropriate to assume the U.S. will lead effective international discussions regarding SB.

It is easy to understand the calls for regulation. For example, the horsepox virus was recently reconstructed by piecing it together from several biosynthesis companies. “This is clearly an example of dual-use research, and observations like these pose significant challenges for public health authorities. Most viruses could be assembled nowadays using reverse genetics, and these methods have been combined with gene synthesis technologies to assemble poliovirus and other extinct pathogens like the 1918 influenza strain. Given that the sequence of variola virus has been known since 1993, our studies show that it is clearly accessible to current synthetic biology technology, with important implications for public health and biosecurity.” [165]

Another recent SB project funded by the U.S. Defense Advanced Research Projects Agency (DARPA) raised dual-use concerns because it investigates whether insects can more rapidly disperse genetically modified viruses to alter crop genomes [178]. Although the stated purpose of the research is to develop better food security against bioweapons, some researchers note that the genetically modified viruses spread by insects could actually alter crops in malicious ways. “[E]asy simplifications (and not elaborations) of the described work program could be used to generate a new class of biological weapon.” [178]

If the public perceives a genuine threat that SB can facilitate recreation of previously-eradicated, deadly, and infectious viruses, then policymakers might feel pressured into reactionary approaches, such as a moratoria on SB research or a ban on the use of federal money to recreate known deadly viruses. The problem with reactionary regulations is that they are either overbroad or underinclusive, often featuring poorly crafted definitions or omitting key definitions entirely.

There have been a few hard-law initiatives which demonstrate how they can be underinclusive. In 2010, the Department of Health and Human Services (DHHS) issued what was considered a watered-down set of guidelines for screening DNA sequence orders, guidelines which suggested faster and cheaper processes compared to what private industry had proposed [150]. The 2001 PATRIOT Act criminalized the possession of biological agents, toxins, and delivery systems, and the 2002 Agricultural Bioterrorism Protection Act required background checks and registration with the DHHS for anyone in possession of designated Select Agents [150]. However, DNA sequences can be slightly altered or divided into multiple segments, making it unclear how federal statutes apply to attempts to assemble a virus from multiple vendors.

In 2004, the Executive Branch established the National Science Advisory Board for Biosecurity (NSABB) under the National Institutes of Health (NIH), which offered hope that the federal government was taking a greater role in SB regulation, but that role seems to have resulted in “prudent vigilance” and “responsible stewardship.” [150, 159]

Perhaps the most promising efforts came from synthetic biologists themselves, working to establish frameworks for self-regulation at conferences called SB1.0 and SB2.0, held in 2004 and 2006, respectively. The SB2.0 conference discussed several calls to action:

1. Insist That All Commercial Gene Synthesis Houses Adopt Current Best Practice Screening Procedures;

2. Create and Endorse New Watch-Lists to Improve Industry Screening

Programs;

3. Create a Confidential Hotline for Biosafety and Biosecurity Issues;
4. Affirm Members' Ethical Obligation to Investigate and Report Dangerous Behavior;
5. Create a Community-Wide Clearinghouse for Identifying and Tracking Potential Biosafety/Biosecurity Issues; and
6. Endorse Biosafety/Biosecurity R&D Priorities [199].

Unfortunately, SB2.0 failed to produce meaningful consensus, and "SB2.0's failure to act would trigger a cycle of disappointment and low expectations for every SB conference thereafter." [150]

Reactionary regulations can stifle legitimate, beneficial research and make compliance difficult or erect unintended market barriers. In fact, a strong reactionary approach could have prevented development of the recent promising treatment for pediatric brain tumors through deactivated strains of synthesized poliovirus [204], or the development of safer and affordable insulin [81]. Continued and open research in SB is one of the most promising avenues for detecting, understanding, and responding to outbreaks of diseases [62].

Indeed, the European Court of Justice in July 2018 declared genome editing, which is a technique commonly used in SB, to be essentially the same as traditional genetic engineering and thus theoretically subject to all of the same existing GMO regulations. One researcher claimed that "[t]he classification of genome-edited organisms as falling under the GMO Directive could slam the door shut on this revolutionary technology. This is a backward step, not progress." [180] Another researcher claimed "the decision could set agbiotech in Europe back another 20 years." [180]

Interestingly, the FDA recently approved a new drug for treating smallpox, despite the World Health Organization having declared the disease eradicated in 1980, because “there have been longstanding concerns that smallpox could be used as a bioweapon.” [66] Although the FDA did not mention SB in its approval for the drug, it is safe to assume the world needs such drugs as a direct result of SB dual-uses, and in turn those drugs could be manufactured with the help of SB techniques.

Given this background, the rest of this chapter will argue why there is no appropriate or sufficient national or international form of hard law for regulating SB. Despite failed prior efforts to establish best practices for SB safety and security [150], it is more important now than ever to renew the quest for meaningful forms of oversight. The best approach is a soft-law framework between government, academia, industry, the DIY community, and the public. Lastly, we encourage SB researchers to incorporate software engineering principles as a way to validate and verify the safety and security of their designs.

5.1 Comparisons and Contrasts with other Dual-Use Technologies

Within one year of the discovery of X-rays in 1895, several papers noted their potential to cause harm to life and the environment [40]. Since 1928, the International Commission on Radiological Protection (ICRP) has been making recommendations regarding safe exposure to radiation [106]. The ICRP is a voluntary international organization with no governmental power, so it can only encourage nations to adopt regulations to enforce its recommendations.

A similar organization could be of great benefit for addressing SB risks. However, radiation levels and exposures are relatively easy to measure, as well as quantifying which levels and exposures are safe to encounter. It is a completely different and much more difficult task to determine whether SB designs are safe or could become dangerous through mutation or evolution. “While it is feasible to utilize PCR [polymerase chain reaction] to identify a Select Agent pathogen ‘needle’ from the enormous environmental background ‘haystack,’ there is no technology available today that can reliably alert us when a novel pathogen, whether natural or engineered, is present in the environmental background.” [156] Indeed, because “engineered microorganisms are self-replicating and capable of evolution, they belong in a different risk category than toxic chemicals or radioactive materials.” [208]

With regard to nuclear weapons, there are many bilateral and multilateral treaties in place for control and non-proliferation. Similar treaties could be helpful for control and non-proliferation of SB-based weapons, but the process of crafting, negotiating, and ratifying a meaningful multilateral treaty would likely take decades for several reasons. First, definitions are critical in treaties, and “there remains no universally agreed upon definition of [SB] . . . .” [156] This threshold definitional question also implies that current treaties banning chemical or biological weapons are inapplicable to SB-based weapons. “With regard to chemicals, biochemicals, and toxins, synthetic biology blurs the line between chemical and biological weapons.” [156]

Second, any multilateral treaty would have to include enforcement and compliance mechanisms. Enforcement of international treaties is always difficult and complicated regardless of subject, but compared to SB it is relatively easier to determine compliance with weapons treaties compared to whatever current treaties apply to SB. That is because the knowledge, infrastructure, materials, and overall physical footprint needed to refine uranium into fissile material are gargantuan compared to what is needed to engineer *E. coli* cells to produce known pathogens [218].“The production of most DNA viruses would be achievable by an individual with relatively common cell culture and virus purification skills and access to basic laboratory equipment, making this scenario feasible with a relatively small organizational footprint (including, for example, a biosafety cabinet, a cell culture incubator, centrifuge, and commonly available small equipment).” [156] Furthermore, “activities requiring less equipment may be able to be pursued by actors with fewer resources and may be conducted in a clandestine laboratory, making detection or attribution more difficult and therefore making concern higher” and “it would be extremely difficult, if not impossible, to distinguish a facility being used to develop bioweapons based on synthesized pathogenic bacteria from a legitimate academic or commercial facility.” [156] This means SB-based weapons can be produced by not only technologically advanced nations with abundant resources, but also by small terrorist groups or even an individual with malicious intent, making enforcement of and compliance with an international SB treaty either pointless or impossible, depending upon a nation’s perspective.

The Internet and communications networks by themselves represent a dual-use technology, because the flow of information can lead to good or evil—regardless of whether the intent is to share a recipe for chocolate cake or uranium cake. Internet and network security generally cannot be guaranteed because it is impossible to prove software is “correct.” [57] Thus, attackers will continue to have the upper hand because of: zero-day exploits; a generally open network architecture; and heavy technical debt associated with complicated interfaces between the Internet’s layers and protocols. For a recent example, the McAfee Advanced Threat Research team “discovered that access linked to security and building automation systems of a major international airport could be bought for only US$10.” [71]

A good (or accidentally bad) programmer can create malware or a virus that spreads across the Internet [166], just as SB engineers can create a virus that spreads among people [165]. Good “cybersecurity,” another term without an agreed-upon definition, is currently a set of best practices and penetration testing. Perhaps SB safety and security could benefit from something like the NIST Cybersecurity Framework, a voluntary set of “standards, guidelines, and best practices to manage cybersecurity-related risk.” [157] Developing such a framework for SB would likely take many years, and the task would be beyond the mission and capabilities of any single U.S. agency. Thus, an SB framework is best relegated to a voluntary international organization of experts similar to the ICRP.

As a final technological comparison with SB, consider the concern over 3D printing of guns. The ability to purchase or share plans to print plastic guns at home is comparable to SB and its DIY movement [89]. Just as the files and materials needed for 3D designs are easily purchased, shared, or available from open-source repositories, so too are DNA sequences and the reagents needed to implement SB designs. Sequences for thousands of toxins and pathogens have been available online from open-source databases for years, and even the iGEM BioBrick database includes a sequence for the invasin gene which includes the ominous statement: “makes *E. coli* a good bioweapon by allowing bacteria to enter and live inside human cells.” [100] At this point, any attempts to restrict access to DNA sequences, equipment, and materials would be similar to confiscating 3D printers and asking owners to delete all associated data.

Having described some fundamental similarities and distinctions between SB and earlier dual-use technologies, the next two sections will address what makes SB unique and so difficult to effectively regulate through hard law.

5.2 Biohackers and DIY: Unknown-Unknowns?

Unlike radiation, the Internet, or 3D printers, SB allows anyone with $2,000 to manipulate DNA, RNA, and its flow within and without cells. A DIY SB kit is currently available for purchase for under $2,000, which “provides all the equipment, reagents and materials you need to get started in molecular biology and genetic engineering,” [202] including a centrifuge add-on, shipping not included. There are even cheaper kits for $150, which are targeted for middle-school science classrooms and home use which “let you edit a bacterial gene using instructions made for those without expertise in little more than a weekend [56]. In 1970, Francis Crick referred to the manipulation of the flow of DNA and RNA as the central dogma of molecular biology [46]. In other words, anyone with $2,000 can now manipulate and program life as if a cell were hardware and DNA were software [201]. (One could argue radiation allows manipulation of DNA through mutation, but SB allows programmed manipulation as opposed to stochastic mutation via overexposure to X-rays.)

“Drawing on a computer science imaginary, synthetic biologists compare themselves with Silicon Valley garage ‘tinkerers’ and ‘hackers’ who sparked the growth of the microprocessor industry in the 1970s.” [75] If DIY SB kits were limited to relatively benign projects such as green-glowing plants [128], maybe they would find a place in history alongside Ant Farms [243], Sea-Monkeys [228], or the Gilbert U-238 Atomic Energy Laboratory [227]. However, with time and progress, what experts with plentiful financial backing can achieve today will become cheaper and easier for the DIY SB community. “Many DIY biology activities are expressly educational, fun, or tied to local community needs (for example, testing food samples). Yet while most of these DIY projects are not sophisticated, the model does make accessible to the general public tools that can be used to do advanced work,” and although “iGEM projects are carried out by students, many of them entirely new to bioscience, some projects have been quite sophisticated.” [156]. To its credit, the iGEM competition heavily stresses biosafety and biosecurity [269].

The most famous DIYer from the SB community (or “biohacker”) has been Aaron Traywick, who notoriously injected himself live online with what he claimed was his own SB cure for herpes [28]. Traywick died months later from unrelated circumstances, having drowned in a sensory-deprivation tank [27]. Traywick and his company Ascendance Biomedical offered a presale campaign for DIY gene therapy kits, to which the FDA responded: “The sale of these products is against the law. FDA is concerned about the safety risks involved.” [65] Previously, Kickstarter had already banned campaigns for DIY starter kits based on legal and environmental safety concerns with the Glowing Plant Project [104].

Josiah Zayner, a rival colleague of Traywick and founder of The Odin, currently sells DIY SB kits and supplies and “is working on creating DIY experiment kits made for use with frogs, so that people can practice science more safely, and not move straight to experimenting on themselves.” [28] Zayner, who has also injected himself with what he claimed to be DNA to modify his muscle genes to give himself bigger muscles [132], has said DIY SB injuries are only a matter of time: “There have been people who’ve contacted us for the sole purpose of buying stuff from us to inject. We seriously discourage people from doing that. Obviously we can’t stop them from doing it, but we discourage people and try to point them in correct direction so they can seek out knowledge.” [241] Furthermore, and maybe needless to say, “members of the biohacking community may not always be familiar with the due biosafety procedures” and “[c]oncerns include accidental release from one without proper training in laboratory safety protocols as well as bioterrorism.” [2] In 2018, a 16-year-old high school student from France claimed to have injected himself with a DNA sequence based upon an encoding of the Book of Genesis [141].

There is understandable concern, either through intent or negligence, that “a DIY biologist might use genetic engineering or synthetic biology to create and distribute an organism that causes harm, for example by reducing biological diversity, weakening an existing population of plant or animal life, making people sick, destroying crops, threatening the food supply, or deterring tourism.” [104] Zayner has argued, among other reasons, that his DIY kits are safe by analogy with beer: “Currently, almost all yeast used to create beer has not had its genome sequenced. This means that despite the yeast being considered safe by the FDA we have no idea the exact proteins that are inside these yeast[s].” [238]

Zayner has stated he wants to introduce DIY biology to high schools across the globe [126]. This could be a promising venue for building awareness of safety and ethical concerns, but there may be trouble brewing. On May 8, 2019, the California Department of Consumer Affairs notified Zayner that a complaint was filed against him for practicing medicine without a license [126]. On July 30, 2019, California passed SB 180, which requires gene therapy kits to come with prominent notices that they are not designed for “self-administration.” [32] Whether such notices will prevent self-administration remains to be seen.

Although the DIY SB concerns are valid, for the sake of argument, let us assume projects such as tinkering with yeast to make fluorescent beer or lab-made meat [26] are safe, healthy, and an overall benefit to society. The next section addresses the real, current, and legitimate concerns about rogue nations, terrorist groups, or simply determined malicious individuals using SB to manufacture and stockpile known pathogens.

5.3 Malicious Actors: Known-Unknowns?

Biological weaponry dates back to at least ancient times, with the Assyrians poisoning wells with a fungus in the sixth century BC [125]. In 1346, the Tatar army catapulted corpses contaminated with the Black Death at Genoese sailors [135]. With the advent of the Biological Weapons Convention in 1972, “the development of pathogens as weapons became the province of clandestine nation-state programs and non-state actor terrorism.” [156] In 2016, U.S. director of national intelligence, James Clapper, asserted that emerging SB threats “should be listed as dangers alongside nuclear tests in North Korea or clandestine chemical weapons in Syria.” [160] In 2012, Vladimir Putin said that Russia would be pursuing new genetic high-tech weapons systems which “will be comparable in effect to nuclear weapons but will be more ‘acceptable’ in terms of political and military ideology.” [242] At the 2016 World Economic Forum in Davos, a number of experts expressed particular concern that non-state actors could “effectively weaponize and disseminate biological agents.” [210]

5.3.1 Twenty Years’ Warning: The Bioterrorism Threat to Public Health

As far back as 1998, researchers have sounded the alarm over the ever-present danger of malicious actors deploying biological weapons in terrorist attacks [94]. “Nations and dissident groups exist that have both the motivation and access to skills to selectively cultivate some of the most dangerous pathogens and to deploy them as agents in acts of terrorism or war.” [94] Furthermore, biological terrorist attacks present a variety of technical issues that make them especially difficult to detect, respond to, and attribute.

If a malicious actor has the ability to “reboot” the smallpox virus, it would have the potential to cause widespread casualties and panic because most of the world’s population no longer receives the smallpox vaccination. “In the modem world . . . smallpox is almost certainly the only pathogen capable of inflicting large-scale casualties. The idea that terrorists could deliberately infect themselves to spread the disease was already widely known at the start of the twentieth century.” [150]

Only a small number of specific groups receive the smallpox vaccine today, such as hospital workers, laboratory researchers, and military personnel. In 1998, it was estimated that only 10-15% of the world's population retained any smallpox immunity, and the mortality rate of the unvaccinated who contract the disease is typically between 25-30% [94]. There is no known specific treatment for smallpox, and regardless, modern cities would not have nearly enough hospital beds to handle a major outbreak [94].

Even if there were adequate hospital beds and an effective treatment for smallpox, health organizations would not know of a potential outbreak, because those infected show no signs during the ten to twelve-day incubation period [94]. During this time, those infected are contagious [94]. When the first symptoms arrive, it could still take several days for medical professionals to recognize a smallpox outbreak, because of the understandable lack of familiarity with the disease's symptoms such as high fever and pain. Indeed, during an outbreak in 1972 in Yugoslavia, one patient died before an accurate diagnosis, and was buried two days before the first case of smallpox was recognized [94]. In all, it took four weeks after the first patient exhibited symptoms to make an accurate diagnosis of smallpox. By that time, 150 people were infected [94], resulting in a total of 175 persons infected with 35 fatalities [94].

Lastly, even if there were plenty of lead time to adequately detect and respond to an outbreak, it would be extremely difficult for any nation-state to prove attribution for a biological attack. This is especially true if that nation-state wishes to provide legal justification for any type of formal retaliation against another nation-state [156].

There are many viewpoints and uncertainties with respect to how likely, severe, and possible malicious or negligent SB designs are or could be in the near future. This could reflect hindsight bias, or the idea that because we have yet to see a clear example of SB used in an attack, such attacks are either not possible or not as easy to accomplish as some have warned.

At the worst extreme, there are doomsday-like arguments that unchecked SB could lead to an existential threat to humanity or other species and the environment [24]. At the other extreme, some argue that serious SB weaponry would require too many resources, raise too many suspicions, be prevented through market forces, and ultimately entail modifying organisms to the point where they are "too delicate to survive in the open." [150]

Somewhere between those two extremes is the 2018 NAS report which analyzed what threats would likely look like in the near future [156]. The report concluded that the most probably threats would stem from "re-creating known pathogenic viruses, making existing bacteria more dangerous, and making harmful biochemicals via in situ synthesis." [156] Under those scenarios, the most dangerous plot would likely entail a rebooting of smallpox, making it more virulent or communicable, and sending infectious "volunteers" to major transit hubs concurrently around the world.

5.3.2 Example from the Past: The Aum Shinrikyo Cult

The threat of biological terrorism is a very real and probable threat—likely to be ever-present yet be hiding in plain sight. Of particular concern are non-state actors, because they are "the most difficult to detect, deter, or regulate." [74] An illustrative warning example comes from a detailed 2011 report on the Aum Shinrikyo cult, which is most infamous for its sarin gas attack on the Tokyo subway system in 1995 which killed 13 people and hospitalized approximately 6,000 more [237]. The cult had several facilities, including one which produced the sarin gas used in the Tokyo attack, which was built for an estimated $30,000,000 in 1993. This building far exceeded the amount of space and the estimated $20,000 it would take to build an SB DIY garage today [182]. The cult used shell corporations to distract authorities from learning its true function, but there were several other red flags that should have spurred closer scrutiny of the cult's activities at its various facilities.

The report highlights several alarming features about the cult's success, despite their relatively modest resources and limited technical knowledge. Although the cult originally failed at creating and weaponizing biological weapons such as botulism and anthrax, they had much more success when they switched to production and distribution of sarin gas, a chemical weapon. One of the reasons the cult abandoned its pursuit of biological weapons was that in the early 1990s, it was fairly difficult to perform the necessary biological laboratory work to produce enough materials for an attack. Despite several technical blunders, in-fighting, and the pursuit of scientifically "bizarre ideas," the cult's determination resulted in the production of enough sarin gas for several attacks, as well as "research quantities" of: LSD; sodium pentothal; PCP; methamphetamines; GF nerve agent; VX nerve agent; gunpowder; barbiturates; mescaline; and catalysts for ammonium and nitric acid [237].

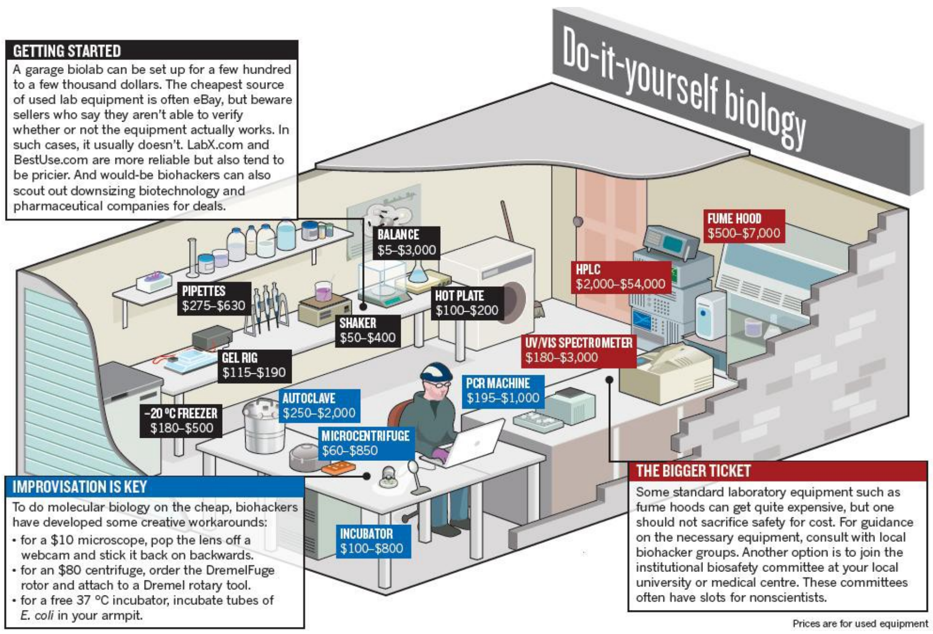
The successful chemical weapons program is concerning for many reasons. One, there were only a handful of technical experts on hand who knew how to produce biological and chemical materials. Two, they failed several times and attracted the attention of various authorities who never fully discovered what the cult was plotting, even though the facilities had relatively large footprints and caused nearby residents to become suspicious. Three, the facilities were relatively affordable to build and avoided government oversight through legal obfuscation.

The 2011 report concluded: "Terrorists need time; time will be used for trial and error (tacit knowledge acquisition); trial and error entail risk and, in this case, provoked disruption; but Aum found paths to WMD, and other terrorists are likely to do the same . . . . Groups such as Aum expose us to risks uncomfortably analogous to playing Russian roulette. Many chambers in the gun prove to be harmless, but some chambers are loaded.” [237].

As will be seen in the next subsection, a modern-day rogue group would likely have a much easier time than Aum Shinrikyo if it chooses to pursue a biological weapons program. First, the number of people skilled enough in SB to perform the laboratory work needed is growing and the technology of SB itself is becoming more “de-skilled” and “democratized.” Second, the facilities needed for that laboratory work require much smaller physical footprints. Third, the facilities and equipment needed for that laboratory work are much cheaper (and getting cheaper). Instead of $30,000,000 for a chemical factory, a fully-functioning SB lab now costs around $20,000 (see Figure 5.1) and can fit in a one-car garage. Additionally, it would be even easier to cover up through legal misdirection, the appearances of “legitimate” research, and the use of cryptocurrencies to purchase materials.

Even back in 1998 one researcher asserted: “biological terrorism is more likely than ever before and far more threatening than either explosives or chemicals.” [94] “With the advent of communities like LabX, which sell and auction used lab equipment suggests that someone may even be able to order a machine with little oversight and monitoring of who the recipient is in the first place.” [56]

Today, malicious actors have many more methods to acquire biological agents than the Aum Shinrikyo Cult. Since 2011, the “[t]echnological barriers to acquiring and using a biological weapon have been significantly eroded.” [210] A malicious actor could “isolate starting material from an existing environmental or laboratory source, request material from a peer or repository, purchase pre-made sequences from a repository or commercial source, purchase custom synthesis, or use his or her own (or his or her own institution’s) synthesizer.” [74] The



GETTING STARTED

A garage biolab can be set up for a few hundred to a few thousand dollars. The cheapest source of used lab equipment is often eBay, but beware sellers who say they aren't able to verify whether or not the equipment actually works. In such cases, it usually doesn't. LabX.com and BestUse.com are more reliable but also tend to be pricier. And would-be biohackers can also scout out downsizing biotechnology and pharmaceutical companies for deals.

IMPROVISATION IS KEY

To do molecular biology on the cheap, biohackers have developed some creative workarounds:
- for a $10 microscope, pop the lens off a webcam and stick it back on backwards.
- for an $80 centrifuge, order the DremelFuge rotor and attach to a Dremel rotary tool.
- for a free 37 °C incubator, incubate tubes of *E. coli* in your armpit.

THE BIGGER TICKET

Some standard laboratory equipment such as fume hoods can get quite expensive, but one should not sacrifice safety for cost. For guidance on the necessary equipment, consult with local biohacker groups. Another option is to join the institutional biosafety committee at your local university or medical centre. These committees often have slots for nonscientists.

Figure 5.1: Estimated costs for equipment to establish a DIY SB garage in 2019 [182].

cheapest and arguably least-challenging method would be to order a known toxic sequence divided into several different chunks from several different synthesis companies. The individual orders are unlikely to trigger red flags during the screening process, and they can be reassembled into the original sequence using the Gibson Assembly method [88, 224]. Further obfuscation of malicious intent is possible by padding unnecessary DNA sequences to the beginning and end of the target sequences [74]. Even more worrisome, "it will soon be possible to clone the harmful genes into a harmless bacterial vector to produce the toxin of interest autonomously. Currently, the synthesized gene can be shipped to the actor, who may then choose to produce the toxin using commercially available bacteria and few other reagents and tools." [56]

5.3.3 Today's Ongoing Threats

Ten years ago, it was easy to argue that SB threats were years away. For example, it was argued that high-school and undergraduate students lacked the expertise needed to create successful experiments, or at least that the experiments would be so mistake-prone that it would take “many years” to find success [150], or that a biohacker “working with limited resources and less-accurate equipment, would need years to re-engineer and weaponize a drug-delivery bacterium.” [140] However, since 2004, the iGEM competition has trained tens of thousands of international high-school and undergraduate students to conduct SB experiments. Yes, some of the iGEM team projects result in failures, but there are plenty of examples of successes, and the work is typically done over the course of one summer [101].

Another example argument is that it would be difficult for a terrorist organization to purchase pre-made sequences from existing companies because companies would report suspicious orders to appropriate authorities [150], or that requesting “multiple suspicious parts would trigger alarms.” [140] This argument has already been weakened by two specific examples: the purchase of the smallpox virus [2] and the reassembling of the horsepox virus [165].

There are also many more gene synthesis companies today, internationally, making it easier to obfuscate intent by purchasing smaller sequences to be assembled after delivery. For example, BGI Genomics has offices in the U.S., the U.K., Denmark, Thailand, China, and Australia. It was once argued that the U.S. would maintain its relative dominance through economies of scale and we would not need to worry about regulating companies beyond the reach of U.S. law [150].

Furthermore, equipment is now available for $800 which can synthesize sequences of 1,200bp, meaning malicious actors could bypass private synthesis altogether [203]. The existence of such affordable equipment defeats another old argument: that affordable desktop devices would not be an imminent product for purchase because synthesis companies would try to protect their market share by investing in large-scale synthesis plants [150].

Lastly it has been argued that because the world has not seen a nuclear attack since 1945, this is evidence to suggest terrorists will be unable to successfully avoid regulatory schemes to implement a successful SB attack [150]. However, there are, and have been, several fundamental differences between developing nuclear weapons and developing SB weapons, although both represent “existential risks” to humanity [24]. Nuclear weapons require a large physical footprint and access to heavily restricted materials and machinery, and tests are relatively easy to detect because radioactivity is measurable. Or as one commentator has noted: “We were lucky that making nukes turned out to be hard.” [25]

On the other hand, experimenting with SB requires little more than a garage filled with materials and machines which are relatively cheap and easy to acquire, and they do not emit radiation, let alone malicious intent. Yes, a malicious actor would need some safety mechanisms to ensure technicians are not exposed to deadly materials, but terrorists are dedicated and fanatical, and they are not as likely to be concerned about safety or security as would industry or academia.

In 2018, the National Academies of Science issued a report titled “Biodefense in the Age of Synthetic Biology.” [156] The report assessed which national security threats were most worrisome and most likely given the current state of affairs as well as within the next five years. The report concluded that the most immediate threat, borrowing a term from computer science, was the “booting” of existing pathogens. For example, a malicious actor could use the relatively safe strain of *E. coli* and use SB techniques to make a cell (the chassis) produce pathogens for which the DNA sequences are already known. “Rapid advances in DNA synthesis technology have made it possible to obtain a pathogen without direct access to the infectious agent itself. Today, any viral genome can be synthesized based on published sequences, and booting that sequence into a replicating form is also feasible for most viruses.” [156] The report also asserted two other immediate concerns: “making existing bacteria more dangerous, and making harmful biochemicals via in situ synthesis.” [156]

The report gave several reasons for these immediate concerns. First, SB makes it easier, cheaper, and faster for a growing number of malicious actors to manufacture biological or chemical weapons. The equipment and materials needed are increasingly accessible, and DNA sequences for known pathogens are online in public databases. Not only could malicious actors boot a pre-existing virus, they could experiment with its characteristics to make it more communicable, more harmful, or more difficult to detect [237].

Furthermore, a malicious actor does not need to understand the underlying biology, chemistry, or genetics to create an effective weapon. “Similar to the way that modern programming languages do not require software developers to understand how software routines are executed at the transistor level, biological design tools are becoming less dependent on base-pair level descriptions of genetic constructs. In other words, a synthetic biologist may not need to know the exact sequence of nucleic acids required in order to design a regulatory circuit for gene expression–simply specifying a particular goal . . . may be sufficient . . . .” [156]

Second, people who are willing to use SB to create weapons are not likely to care about being safe or perfecting their designs. “It is also conceivable that malicious actors would forego some of the rigorous testing that other researchers would perform, since the standard of success–creating an agent capable of doing ‘enough’ harm–is markedly different from the standards involved in publishing results in a scientific journal. Malicious actors may also be able and willing to test in human subjects, unhindered by the moral considerations and ethical frameworks that guide other research efforts.” [156]

Third, SB requires relatively little physical space, such as a basement or garage, making it extremely difficult to monitor or discover malicious activity. “High potency molecules that can be produced through simple genetic pathways are of greatest concern, as they could conceivably be developed with modest resources and organizational footprint.” [156] Implicitly, low-potentcy molecules could nevertheless form the basis of an effective attack, and SB weaponry would not necessarily require sophisticated deployment techniques if malicious actors find willing hosts or if existing pathogens are engineered to be more transmissible.

Finally, attributing SB attacks to their source will be extremely difficult, if not impossible, similar to the attribution problems which plague forensics experts in cybersecurity. “The increasing physical and virtual separation of design and manufacturing not only further increases the accessibility of synthetic biology but also creates potential security concerns where designs cannot necessarily be explicitly connected to manufacturing locations and vice versa.” [156] Attribution will also be difficult because by far most of the equipment and materials are used for beneficial purposes and general applications, so detecting suspicious activity through individual purchases or any one individual’s online behavior could be ineffective, especially as SB participation increases around the world. “[A]ttribution in the age of synthetic biology is likely to be heavily dependent on computer-based approaches that look for molecular signatures, as well as on [traditional methods of gathering] intelligence.” [156]

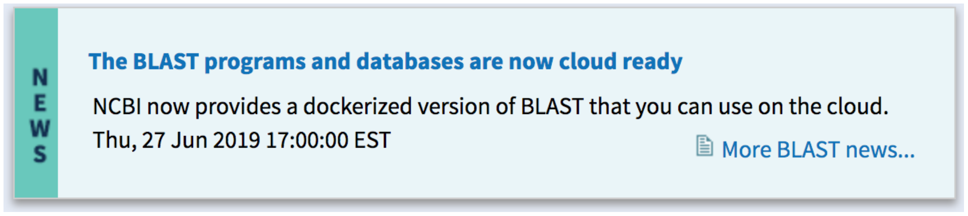
The 2018 report focused on malicious actors using SB to make their own bioweapons, but that is only one facet of the emerging threats. Malicious actors could simply acquire bioweapons through other clever means. A terrorist organization could attempt to compromise employees of DNA synthesis companies, academics institutions, or national governments and simply have them produce dangerous materials to be shipped abroad [171]. Consider that in

2019, two Pakistani men were accused of bribing AT&T employees with over $1,000,000, including $428,500 to just one employee, to install malware to unlock iPhones for use outside of the AT&T network [38]. The 2018 report also does not explore cybersecurity issues associated with DNA synthesis. In fact, researchers have already demonstrated weaknesses in bioinformatics software, and were even able to encode malware into a physical DNA strand [163]. It is thus theoretically possible to mail a physical strand of DNA which could compromise computers connected to sequencing hardware.

Many companies profiting from the SB boom are also not taking security seriously enough. “Despite the many potential risks, there is a surprising level of naiveness among partners in the biotechnology supply chain. This natural trust is partly associated with the perceived reputation of academic institutions or biotech companies.” [171] In the U.S., companies providing gene synthesis services are only encouraged to perform minimal screening of orders against known select agents and toxins, and are not required to perform background checks on their clients [1, 36]. If a company screens orders against known toxins from an online database, the database itself represents an attack vector because it could be compromised by either deleting or modifying entries [163]. It is also possible to manipulate genetic databases to make individuals appear to be related (perhaps for inheritance fraud) or to appear unrelated (perhaps to avoid criminal investigation) [162].

In addition, software and hardware used by SB engineers is not any safer or more secure compared to those used by other disciplines. Iranian hackers have already targeted cloud-based DNA sequencing applications [39]. The motivation could be relatively benign, such as building up a larger botnet for mining cryptocurrency, or it could be more malicious, such as exfiltrating the DNA sequences of specifically targeted individuals for sale on the dark web [39].

A security researcher has also shown that a commonly used “best practice” software package called the Burrows-Wheeler Aligner (BWA) implements unsafe function calls which can result in buffer overflows, allowing an attacker to execute arbitrary code on any genomics machine that runs the software [99]. The BWA software relies on a human genome database stored centrally on one server hosted by the National Center for Biotechnology Information (NCBI) and is vulnerable to man-in-the-middle attacks, which could allow an attacker to manipulate DNA sequencing results, possibly influencing medical diagnoses or forensics analyses [99]. As can be seen in Figure 5.2, the NCBI BLAST tool, which is commonly used to screen sequences against known toxic proteins, is now available through cloud computing services.


NEWS

The BLAST programs and databases are now cloud ready

NCBI now provides a dockerized version of BLAST that you can use on the cloud.
Thu, 27 Jun 2019 17:00:00 EST

More BLAST news...

Figure 5.2: The NCBI landing page showing BLAST searches are available through cloud computing.

If DNA synthesis computers are connected to an Internet-facing network, the ability to run arbitrary code implies that an attacker could simply take control of the synthesis machine and co-opt it to issue any sequence, bypassing the screening process entirely. An attacker could also corrupt the screening databases, whether stored locally or accessed via cloud services, and modify toxin entries to be non-toxic. The attacker could then safely order a dangerous sequence under the assumption it would no longer raise any red flags. On top of it all, sensitive information about BioWatch pathogen sensors, which were installed in 30 U.S. cities as part of the response to the anthrax attacks of 2001, was stored for over a decade on an insecure website [14].

The technical debt surrounding our new “bioeconomy,” which accounts for as much as $4 trillion or roughly 25% of the U.S. GDP annually, has led to some researchers calling for a new hybrid disciplinary called “cyberbiosecurity.” [155] Bioinformatics software and systems suffer from the same vulnerabilities as other “Big Data” processes, and the four steps of the cyber-physical supply chain from 1) sequencing, 2) alignment, 3) screening, and 4) synthesis all face the same vulnerabilities as any other manufacturing industry that sells dual-use technologies [155]. Consider that Big Data, by itself, is a dual-use technology because on one hand it can be used to screen sequences for pathogenic qualities, while on the other it could be used to discover sequences never seen before in nature (“xenonucleic”) which could be more pathogenic than those already known and documented [155]. In addition, the cyberbiosecurity supply chain should be expanded to include aspects of agricultural production and distribution systems, because malicious SB designs could surreptitiously reduce crop yields or destroy crops altogether [74, 155]. Lastly, malicious actors can replace, and thus avoid detection at, any step of the biosecurity supply chain for which they possess adequate technical skill and equipment [56, 74].

All of these threats might seem daunting and insurmountable. The next section offers some hope and suggestions for how to address the emerging threats to ensure SB continues to advance and deliver its full beneficial potential.

5.4 The Inadequacies of International and Domestic Law

5.4.1 The Biological Weapons Convention

Traditional regulatory measures such as treaties, statutes, and regulations are difficult to properly tailor to rapidly advancing technology, and the amount of time it takes to codify them almost ensures they will be outdated upon implementation.

“Unnecessary and unreasonable rules will undermine credibility and respect for the law. Over-regulation can also overburden . . . with a jungle of rules that defeat their own purpose because they are impossible to administer effectively.” [72] Indeed, the Biological Weapons Convention (BWC), perhaps famous for its failures [138], itself explicitly recognizes the dangers of over-regulation, because it requires parties to “facilitate, and have the right to participate in, the fullest possible exchange of equipment, materials and scientific and technological information” so biological agents and toxins can be used for peaceful purposes [210].

However, the BWC lacks universal membership, and it has been criticized for not being sufficiently restrictive. One, “it does not include measures to ensure compliance by States Parties.” [109] Two, it is extremely difficult to determine objectively whether a member is intentionally engaging in malicious activity or legitimate research [109, 211]. Third, it is becoming easy to quickly create pathogenic materials and equally easy to eliminate the evidence, making inspections increasingly ineffective [109] and making allegations easier to deny [211]. Traditional signatures of an illegitimate biological weapons program no longer apply to current SB advances, and it is no longer necessary to stockpile large quantities of pathogenic materials or conduct a weapons program in traditional settings, making it difficult to define which facilities are capable of producing biological weaponry (and thus subject to inspection) [211]. Fourth, the BWC has traditionally concerned itself with larger-scale threats, but it seems that smaller-scale attacks are more likely given the nature of global conflict today, and biological weapons are easier to implement and harder to detect or attribute [211]. Fifth, the BWC lacks universal membership and needs stronger export controls [211]. Sixth, the BWC does not apply to non-state actors, or even individuals, who “have repeatedly demonstrated the intent to acquire and use biological weapons, and their lack of success to date is no guarantee that they might not eventually succeed.” [211]. Perhaps the strongest critique, as expressed by the United States, is that no internationally accepted procedure exists to verify compliance with the BWC’s prohibitions [211].

5.4.2 U.S. Domestic Law

As for domestic law in the U.S., SB has so far been regulated by the so-called Coordinated Framework for Regulation of Biotechnology, which the White House Office of Science and Technology Policy created in 1986. This framework encourages several federal agencies to coordinate with each other and determine which agency has proper authority to regulate specific SB technologies. “The new generation of biotechnologies already does and increasingly will create products that do not fall within the statutory triggers that the Coordinated Framework directed FDA, USDA, and EPA to use to justify regulation.” [172] Another criticism of the current framework is that it focuses too much on the products and uses of SB, but not necessarily the processes or the various access vulnerabilities all along the supply chain [74]. The current framework also focuses on protecting intellectual property and personal medical information, as opposed to sharing information which could be mined for new products and applications [235].

It can be extremely difficult and frustrating for industry leaders to bring new beneficial SB products to market because it is unclear which agency should be petitioned first, which agency will give final authority, and how long the whole approval process will take. The technology is advancing more rapidly than any single agency can adequately handle [74]. Given the potentially wide set of applications, a company could create a new SB crop which: is nutritious food; produces its own pesticide; is a cure for acne; emits red fluorescence if over-watered; and presents a low risk of causing rashes for infants and the elderly. A regulatory compliance attorney is likely to be confused about where to start the approval process, or whether the crop is exempt from agency oversight.

For example, the USDA proposed a new rule for bioengineered food labeling to be codified at 7 C.F.R. part 66, with staggered effective dates starting January 1, 2020 [213]. According to the USDA, the new rule "defines bioengineered foods as those that contain detectable genetic material that has been modified through certain lab techniques and cannot be created through conventional breeding or found in nature." [212] This compact description includes several definitional phrases that will likely require further refinement and interpretation, such as: "detectable genetic material," "certain lab techniques," and "conventional breeding."

There is some guidance for DNA synthesis companies, however, even though the overall supply chain exists in a "minimally regulated environment." [74]. Companies are encouraged by the International Gene Synthesis Consortium (IGSC) Harmonized Screening Protocol to perform background checks on their clients and screen purchased sequences against lists of known pathogens. However, the Department of Health and Human Services (DHHS) does not have any enforcement authority, and there are some exceptions from regulatory oversight from the Environmental Protection Agency (EPA) depending upon which organisms are used in SB designs [74]. In 2005, eight oligonucleotide synthesis companies admitted they did not scan orders regularly, and one company admitted it did not scan longer sequences regularly [4]. At the very least, screening of orders and clients should be mandatory.

However, even these voluntary measures, if strictly followed, are inadequate by themselves. "For example, current guidance did not prohibit a DNA provider from fulfilling an order for the genome of the extinct virus horsepox." [156] In 2006, a journalist from The Guardian ordered fragments of the smallpox gene delivered to his residential address.” [2] Screening can be avoided altogether today, because relatively short sequences can be synthesized in a basic home lab [2]. “An over-reliance on the Select Agent list is a systemic weakness affecting many aspects of the United States’ current biodefense mitigation capability. Another weakness is that companies do not screen DNA sequences of less than 200 base pairs (known as oligonucleotides, or “oligos”). “Screening short sequences produces large numbers of false alarms and remains challenging even today.” [150] This has raised concerns that a determined malicious actor could obtain multiple short sequences from commercial vendors and assemble them to create full-length pathogen DNA.” [56, 156]. For example, the main toxin from black widow spiders is 4,201bp [244], meaning it is theoretically possible to divide the sequence into 22 separate orders and reassemble them without raising red flags.

Some have suggested the need for a “federated” approach among the synthesis companies to coordinate the screening process because current practices “cannot trigger concern when orders are divided up among synthesis companies to obfuscate intention.” [74]. With the “absence of oligonucleotide order screening amongst the industry, there still exist penetrable gaps through the ability to connect them together to form larger genes or even entire genomes.” [56]

The list-based screening is also fundamentally insufficient because it only screens against known-knowns, or those DNA sequences which we know have pathogenic qualities, similar to how anti-virus software typically relies on screening against a known blacklist of malicious software [156]. A similar problem exists for the IGSC Harmonized Screening Protocol which asks members to screen clients against a list of individuals and companies known to have ties to drug trafficking or terrorism [74]. As noted above, it is possible to break up longer sequences into shorter sequences and assemble them after delivery in order to avoid raising red flags. A more cinematic scenario would be for a rogue nation or terrorist group to spend the time and money to send an operative to graduate school to learn microbiology and SB, where all the tools are already available to boot a pathogen under the guise of legitimate research. This highlights the importance of background checks at both the academic and industry levels, and even suggests the need for DIY groups to perform due diligence on their members, in addition to highlighting the need for adequate physical security for the DNA synthesis equipment itself [74].

Within the DIY community, which is generally not subject to federal regulation, the only restraints on malicious uses seem to be self-regulation and an unwritten biohacker code of ethics [2]. Although in 2011, DIY biologists set forth two separate drafts of high-level codes of ethics for Europe and North America [54]. This has led some to suggest DIYers could play an important role as the “glue” with knowledge to share between stakeholders [2]. However, “[c]odes of conduct are only effective when researchers are able to recognize problematic experiments.” [150]

One other hard-law suggestion would be to improve whistleblower statutes to encourage researchers, employees, and members of DIY groups to expose dangerous or suspicious activity. While promising on its face, it is very difficult to determine whether a colleague is conducting dangerous experiments, either negligently or intentionally, and a malicious actor is likely to be more careful to obfuscate their intent [150].

5.5 Hard-law Risks and Soft-law Proposals

Despite these emerging concerns about regulatory oversight for SB, as of 2018, “there is a dearth of proactive rather than reactive initiatives being undertaken to evolve governance structures.” [206] That is not to suggest the international SB community has somehow ignored its responsibilities or to suggest there have been no significant attempts to address SB security concerns. From roughly 2001 to 2010, there were several proactive, well-coordinated, and international attempts to regulate SB security through soft-law and hard-law approaches [150]. In fact, researchers have raised concerns over bio-engineered epidemics since the 1970s [150]. The push for security was stoked by some late 1990s fictional novels featuring engineered biological attacks, the anthrax attacks against the U.S. in late 2001, and an Australian research team which altered some pox viruses to defeat vaccines [150]. Unfortunately, this roughly ten-year effort failed to result in any meaningful security oversight [150].

“The absence of mandatory oversight mechanisms for privately funded individuals may in part account for why they have increasingly come to be seen as a group of concern by a scientific community that has for forty years been granted the authority to self-govern.” [75] Although stronger regulation seems necessary, if only to help prevent an accidentally disastrous SB release, hard laws such as statutes and regulations could easily stifle the research necessary to prevent, detect, and respond to SB accidents or weapons. “[I]t can be difficult to identify a priori which research findings entail dual-use risks . . . scientific freedom and access to information are crucial to technological innovation and . . . restricting publication would slow the development of medical countermeasures against biological threats.” [84] “Metabolic engineering could potentially be used to produce toxins, narcotics, or other products relevant to biodefense. For example, yeast has already been engineered to produce opioids in minute quantities.” [156] Researchers also recently used yeast to unnaturally achieve complete biosynthesis of cannabinoids [143].

Existing U.S. agencies have struggled to neatly square their statutory power with the unique challenges of SB: 1) organisms are living and can mutate and evolve; 2) traditional safety testing techniques could be inappropriate for organisms which have never been seen in nature; and 3) traditional focus on products, not processes, could be inappropriate when it is SB-engineered organisms themselves which are producing the potentially risky end-product chemicals [148].

The U.S. Coordinated Framework is inadequate to handle the rapidly advancing, multidisciplinary applications of SB [2]. One suggestion would be to scrap the framework entirely and ask specific agencies to take lead roles, but that would not eliminate the problem of agencies fighting with each other over their regulatory powers (or lack thereof). Another suggestion has been to require engineers and laboratories to acquire some type of licensing in order to purchase equipment and materials for SB research [115]. This approach seems too late, similar to how it would be nearly impossible to recall all 3D printers and their materials from the stream of commerce. Like 3D printers, SB equipment and reagents have been for sale to the public for years. Nevertheless, it might be possible to identify key materials and activities that trigger suspicion of malicious intent just as regulators have done with the nuclear supply chain [74].

However, smaller companies typically do not view themselves as targets for malicious actors, and the SB industry is marked by fierce competition and the associated pressure to maintain profitability, meaning it will be difficult to foster cooperation. In addition, the "current financial environment in the U.S. has also incentivized research institutions and healthcare facilities to contract genetic sequencing services to Chinese or Chinese-affiliated firms," but those Chinese firms are not subject to the same regulatory oversight with respect to "disease research, health diagnostics, genealogy studies, and personal health information." [235]. "Theoretically, the combination of genetic data through research collaborations, legitimate business agreements, and hacked information being exfiltrated to China would be the largest, most diverse dataset ever compiled."

[235] Moreover, the Chinese government itself may be funding research into gene-edited embryos, an activity which the broader international community views as unethical [175], despite evidence that fertility clinics around the world expressed have interest in offering gene-editing services to their customers. [15] Interestingly, He Jianuki claimed he had followed all of the embryo-editing guidelines from the National Academies of Science, Engineering, and Medicine during the “CRISPR Babies” project [16]. This has led to the International Commission on the Clinical Use of Human Germline Genome Editing to meet and generate “a more detailed, less ambiguous report on embryo editing.” [16] Even if the new guidelines are clearer, unethical experiments will likely continue. For example, China was the location chosen for the creation of human-monkey chimeras “to avoid legal issues.” [48]

Perhaps the most successful soft-law initiative so far has been the IGSC, formed in 2009 as a voluntary industry group dedicated to developing best practices for screening orders and customers. Most of the larger synthesis companies are members of the IGSC which shares alerts when a member receives a suspicious order. A weakness in this process is that a member might not deem an order to be suspicious enough or simply might fail to notify government authorities who could further investigate the order to determine whether it is part of a larger malicious scheme [74]. All companies, regardless of size, have internal pressure to avoid public relations nightmares. That is, which SB synthesis company would gladly make a few extra dollars knowing its products would be used in a terror attack? Even with strong external pressures such as export controls and mandatory disclosures of clients’ intended uses of SB technology, it is relatively easy to obfuscate malicious intent through shell corporations, superficially legitimate descriptions, and the use of multiple brokers to acquire various components [74]. Regardless, biological data is not currently considered “security-related” and not subject to export controls [235].

Also, external pressures would require complete international cooperation to effectively enforce. If one nation allows the sale of SB DIY materials and equipment to any individual, an SB licensing scheme would merely burden those pursuing legitimate research. Or, some nations could be secretly funding dubious research such as engineering human embryos to produce more intelligent human beings [176]. Such research might be considered unethical by international experts but does not otherwise constitute a clear violation of international law.

While hard-law solutions can be extremely difficult to tailor, monitor, and enforce, soft-law solutions lack compulsory enforcement mechanisms and are often not restrictive enough. Moreover, soft-law approaches lack traditional procedural processes which attempt to promote transparency and ensure all stakeholders have a chance to participate [148]. There are also subtle side effects of soft-law regimes, such lack of consumer confidence in the government’s ability to adequately regulate companies, and the lack of stability in what guidelines companies must or should follow [148]. Despite the pros and cons of each approach, a soft-law solution is preferable due to its flexibility, transnational applicability, and potential for iteration to better address frequent and rapid progress.

What the world needs now is a voluntary, non-profit international consortium compromised of all stakeholders to share information and concerns. A concerned citizen should be able to have a productive discussion about SB concerns with a DIYer, an academic researcher, a private innovator, and a government official. The consortium would be responsible for developing and maintaining best practices and codes of ethics, and crafting responses to threats it becomes aware of. There are no substitutes for open communication and knowledge with a technology accessible to anyone with a desk and $2,000. We also need a less adversarial and more cooperative approach for governance between private industry and regulators. The consortium could be guided by the “TAPIC” framework, which was originally proposed to overcome structural challenges to improve public health systems [207].

Other soft-law proposals include “voluntary programs, consensus standards, partnership programs, codes of conducts, principles, and certification programs” with “substantive expectations or requirements.” [148] These more flexible approaches are suitable for a rapidly advancing science, because they can be adopted and amended more rapidly through iteration cycles, can provide a more adaptive oversight system, can more easily cross national boundaries, and are based on a collaborative, rather than adversarial, model [148].

Hard laws are difficult to pass and enact, because “the presence of multiple and decentralized veto-holders in the policy generation, implementation, and reform process makes the development and reform of hard law a complex and politically intensive effort that can take years to decades to achieve.” [207] Or more simply stated, “major statutory or regulatory change is politically difficult, time-consuming, and expensive.” [148] Hard laws could have a similar side-effect as the BWC: pushing SB weapon development into the basements and bunkers of rogue nations and terrorists. Without robust and unhindered counter-research, we can only depend upon our public health systems to “recognize outbreaks of novel pathogens, whether natural or engineered.” [156] However, “the majority of public health experts agree that America is woefully unprepared for any sort of mass-scale bioterrorism event.” [115] We need meaningful cooperation and greater openness.

5.6 Addressing Threats Through Another Dual-Use Technology: Computer Science

Gaps in traditional and soft-law regulatory approaches could be addressed through improving the technology of SB itself, and there are several aspects of applying computer science techniques to SB that could help alleviate some of the concerns regarding dual-uses. There are many parallels between programming life and programming computers, and international efforts to keep SB safe and secure should include representatives from computer science, computer engineering, and cybersecurity.

The rapid advances in SB are unsurprisingly linked to rapid advances in high-speed computing, which itself is also a dual-use technology. The algorithms used in bioinformatics and DNA sequencing are both processor-intensive and memory-intensive, but researchers are developing ways to improve efficiency in terms of both time and physical resources [82]. If computer science has enabled malicious actors to more easily and cheaply create pathogenic cells through SB, then it also should enable researchers to more easily and cheaply detect, predict, attribute, and respond to emerging threats.

Computational theory, artificial intelligence, and systems engineering are increasingly intertwined with SB, and the software tools used to make beneficial designs have the same potential to assist making malicious designs. “The role of computational approaches for prevention, detection, control, and attribution will become more important with the increasing reliance of synthetic biology on computational design and computational infrastructure.” [156] It is important to note that any software tools developed for beneficial purposes are themselves vulnerable to adversarial attacks, especially if running on machines accessible through an Internet connection, meaning a malicious actor could use malware to exfiltrate valuable information from biosynthesis companies or otherwise manipulate systems to achieve control over the design, implementation, and test cycles of a target facility [155, 163, 171].

Synthetic biology can be viewed as programming life, with DNA as code, cells as processing units, and plasmids as functions or methods. As early as 2003, researchers developed the Synthetic Biology Markup Language (SBML) to facilitate international cooperation for modeling metabolic pathways and cell-to-cell signaling [98]. In 2014, the first version of the SBOL was introduced which helped establish a standard for describing composite structures and their interactions with an emphasis on visualization [80]. The SBOL standard integrates with common programming languages such as C++, Java, JavaScript, and Python. There is also a 2018 project called Cameo, which is a Python library for enabling computer-aided design for optimizing metabolic pathways in cellular "factories." [34]

Other examples of applying computer science to SB include: GENOCad, a context-free grammar for developing composite parts from BioBricks [164]; automated cellular circuit design based on three inputs [164]; machine learning to help determine which BioBricks are most likely to be of use to implement specific designs [232]; DNA "barcodes" or "watermarks" for attribution and provenance, along with erecting "genetic firewalls" to prevent unwanted interactions between natural and synthesized organisms [217]; systems control to improve biosafety and robustness in the face of uncertainties related to mutation and unanticipated contexts [49]; and the application of software-engineering principles used to certify and verify safety-critical systems [44, 70], because traditional software engineering principles are a natural complement for synthetic biology as programming life [215]. Good coding practices such as abstraction, composition, low coupling, and high cohesion could be applied to BioBricks, allowing engineers to reason about safety and security earlier in the design process [44, 70].

Drawing parallels between computer science and SB helps illustrate the need for interdisciplinary collaboration, and possibly the need for a common set of definitions to reason about problems and solutions. This means experts from computer science and computer engineering should also be part of any collaborative efforts to promulgate best practices for SB safety and security.

For example, if there is renewed interest in collaborative self-governance through efforts like SB2.0, SB3.0 should include experts from software engineering and cybersecurity to offer different perspectives on what attackers could accomplish on current hardware or software systems, especially if those systems have a publicly-facing network connection. The NSABB could also benefit from considering new members who have in-depth knowledge of hardware and software systems used in the design-build-test cycle.

Practical advice could be simple, such as suggesting that sensitive equipment should be air-gapped from the Internet, or establishing chains of custody for sensitive materials using encryption schemes and “smart” cabinets. From a more theoretical perspective, software engineering principles such as deep learning might help predict whether a novel DNA sequence has the potential to be pathogenic [8].

Other crossover research could be inspired by fault-detection techniques in highly configurable systems, or software certification in safety-critical systems. For example, Mozilla Firefox has over $2^{2,000}$ user-configurable options [198], a number far larger than the estimated number of atoms in the universe (10^{80}) [194], representing a configuration space that could never be exhaustively tested. A DNA plasmid can often be 5,000bp, or $4^{5,000}$ different configurations, a number even larger than $2^{2,000}$. Software testing principles for highly configurable systems such as combinatorial interaction testing [86] could help manage the vast configuration space which DNA represents, with the potential to find biological "faults."

If SB organisms are considered safety-critical systems which can cause harm to life or the environment, then it might be possible to apply validation and verification techniques similar to those used in avionics and the automotive industry. Just as car manufacturers must prove run-time guarantees that there is enough processing power and acceptably low latencies to effectively activate an anti-lock braking system [167], a synthetic biologist could be required to make specific biological run-time guarantees, such as that the engineered organisms will die within 30 minutes after escaping containment [70].

Advances in computer science could also result in "effective counter-measures against actual use of biological weapons," something Japan believes could be a key way to enforce the BWC [109]. Arguably, the primary goal of collaborative research should be to rapidly develop techniques to detect the intentional release of pathogens [158], something which is currently difficult and could take a week or longer [94].

Artificial intelligence could help detect suspicious purchasing patterns, similar to how credit card companies use software to detect fraud. A key innovation would be to scan purchase orders place across multiple biosynthesis companies to determine whether those individual orders could be assembled into a pathogenic construct. To be sure, this is a lofty goal. First, it would likely require intelligent, connected labs [56] and a third-party database for tracking customers and purchases. Second, collaboration is nevertheless unlikely among companies in a highly competitive market where false-positives could lead to the loss of legitimate clients [74]. Furthermore, there will always be tension between a client's desire for confidentiality and a regulator's desire to access information, especially if the client wishes to protect DNA sequences as trade secrets [74].

Despite these pragmatic concerns, the U.S. has suggested the need for an interactive database for sharing information and facilitating requests for assistance [211]. All of these research directions hold great promise for improving safety and security of synthetic biology, with the underlying insight that complex problems have complex solutions.

5.7 Conclusion

Although this chapter serves as a warning about current SB risks, it hopefully makes clear that reactionary regulations would stifle beneficial research. Of particular concern would be hard law preventing the type of research needed to counter malicious uses of SB. Malicious actors are unlikely to concern themselves with legal, ethical, moral, safety, or security concerns. A much better approach would be an international non-profit organization based on soft-law principles of openness and cooperation. Computer Science principles will continue to play a vital role in the development and effectiveness of SB applications.

Chapter 6

Extending Existing Methods for Toxin Prediction

With the advances in DNA next-generation sequencing (NGS) and high-speed computing, it has become possible to analyze vast amounts of genomic data in theoretically infinite ways. Recent research has focused on applying supervised machine learning techniques such as natural language processing (NLP) or support vector machines (SVMs) to ascertain meaning from DNA sequences [77, 107]. A typical goal is to predict protein characteristics or functionalities based on DNA sequences which have known properties, such as whether a sequence represents a potentially cancerous mutation or whether it is a toxin or otherwise pathogenic.

However, it is difficult to find meaning in DNA sequences without context because they are not necessarily tokenizable like natural languages. It is also difficult to know the function or characteristics of novel proteins if there are no analogs available for referencing against annotated protein databases.

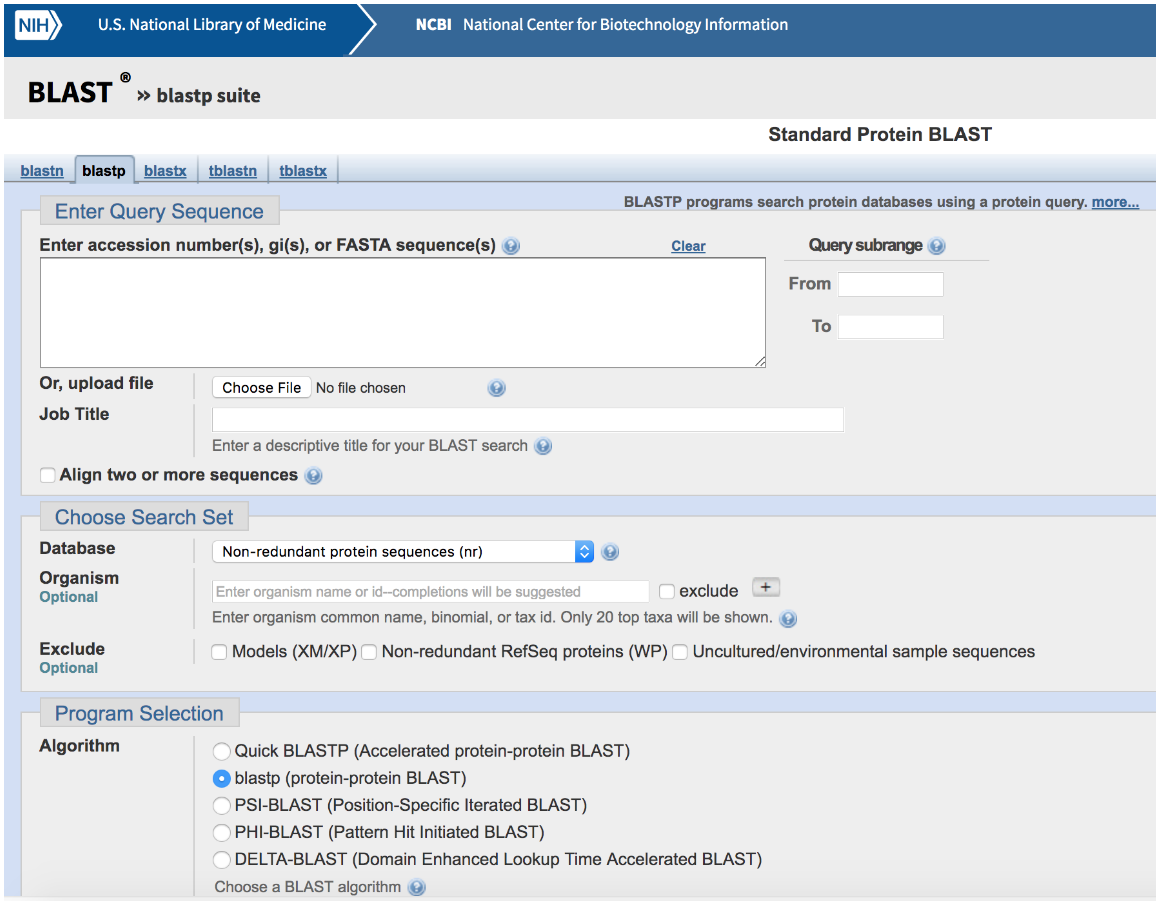
Other recent research has placed emphasis on the importance of microRNA strands (roughly 22bp) and their effect on gene expression and mutation [13, 147]. Researchers have tried to determine which motifs, or patterns, appear more or less frequently in beneficial or harmful microRNA strands.

This chapter, inspired by revisiting an early text-compression algorithm [78], presents a way to find and visualize motifs which are common (or uncommon) in known toxin and non-toxin sequences, with the aim of providing in silico guidance for determining characteristics and functionality of novel synthesized proteins. The result is an extension of existing tools to provide more granularity of information and new visualizations for training data sets.

6.1 Motivation

It is possible to synthesize arbitrary DNA sequences by ordering them from an increasing number of DNA synthesis companies with names like Genscript [248], Gene Universal [247], and Twist Bioscience [281]. It is not necessary to outsource shorter sequences, because they can be synthesized by personal lab equipment in a garage [2]. A primary concern over the ease and affordability of DIY SB is whether malicious actors could order supplies of known toxins, the sequences of which are available in annotated, open-source databases such as UniProt [7]. There are no formal legal requirements for DNA synthesis companies to screen their orders for toxicity or pathogenicity, nor for them to perform background checks on their clients [1]. They are encouraged to perform background checks on clients and screen requested sequences against lists of known toxins and pathogens, but these voluntary measures are inadequate by themselves, even if strictly followed. Over-reliance on screening against known toxins leads to a false sense of security [74], similar to how anti-virus signature lists do not protect against zero-day exploits.

Figure 6.1 shows the main online interface for BLAST [245], a tool which allows anyone to upload DNA sequences to compare their similarities with a large database of annotated sequences gathered from nature. BLAST is maintained by the National Center for Biotechnology Information and is the most commonly used for screening sequences. There are several different string-distance and alignment algorithms from which to choose, as well as 27 databases from which to search against. A user can search against all non-redundant entries, or any subset of 27 databases. Although the algorithms are deterministic, results can vary depending upon which databases are used. Ideally, DNA synthesis companies will use BLAST to screen every order to confirm customers have not requested a known toxin or pathogen.



Enter accession number(s), gi(s), or FASTA sequence(s)

Or, upload file: No file chosen

Job Title: Enter a descriptive title for your BLAST search

Align two or more sequences

Database: Non-redundant protein sequences (nr)

Organism (Optional): Enter organism name or id--completions will be suggested; exclude

Enter organism common name, binomial, or tax id. Only 20 top taxa will be shown.

Exclude (Optional): Models (XM/XP) Non-redundant RefSeq proteins (WP) Uncultured/environmental sample sequences

Algorithm:
- Quick BLASTP (Accelerated protein-protein BLAST)
- [✓ blastp (protein-protein BLAST)]
- PSI-BLAST (Position-Specific Iterated BLAST)
- PHI-BLAST (Pattern Hit Initiated BLAST)
- DELTA-BLAST (Domain Enhanced Lookup Time Accelerated BLAST)

Choose a BLAST algorithm

Figure 6.1: The online NCBI BLAST tool for comparing a sequences against known proteins [245].

However, DNA sequences of less than 200bp, called “oligonucleotides” or “oligos,” are not screened [156]. “This has raised concerns that a determined malicious actor could potentially obtain multiple short sequences from commercial vendors and assemble them to create full-length pathogen DNA.” [156] Furthermore, better screening techniques could benefit the growing DIY community, as the equipment becomes cheaper and more accessible for individuals to synthesize their own oligonucleotides [2].

Another challenge is that list-based screening is fundamentally insufficient because it only screens against toxins which researchers have confirmed have pathogenic qualities [156]. Moreover, list-based screening can produce false-positives by flagging harmless proteins as dangerous for being homologous to known toxins [161]. Because SB technology allows any sequence to be synthesized, even those which have never been seen before in nature, it is critical to be able to reason about novel sequences to determine whether they might be harmful before they are released into the environment [161].

There has been much research devoted to applying NLP techniques to DNA sequences to predict protein structure, function, and pathogenicity [107, 236, 239]. Although promising, "one of the apparent differences between biological sequences and many natural languages is that biological sequences (DNA, RNA, and proteins) often do not contain clear segmentation boundaries, unlike the existence of tokenizable words in many natural languages." [8]

Despite the challenges, there seems to be a growing consensus that the frequency of DNA motifs is, among others, a valuable data point for determining protein characteristics. The intuition for this chapter is that analyzing frequencies of longer motifs will provide additional data points and greater granularity of input for protein classifiers. If microRNA strands of roughly 22bp are important for determining protein characteristics, then tools should be able to provide information about motifs of at least that length, but state-of-the-art tools do not.

To that end, this chapter presents an easily implementable algorithmic approach to gather basic statistics about DNA sequences generally, but implemented with two main groups: 1) a set of known toxins; and 2) a set of known non-toxins. The core idea is byte-pair counting for frequencies of character strings. The data is then visualized in a variety of ways which could be used to establish training sets for other machine learning prediction tools.

6.2 Related Work

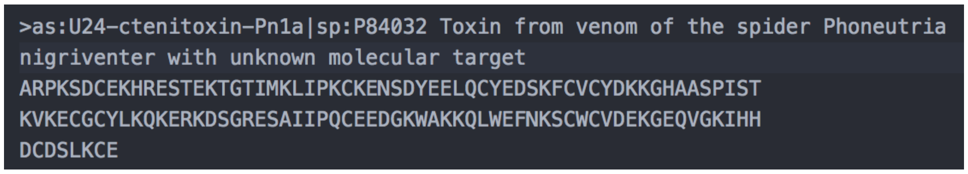
Over the last 20 years, there has been increasing emphasis on the importance of microRNA strands in facilitating gene expression and DNA translation [13, 147]. MicroRNA strands are roughly 22bp in length, representing 4^{22} or roughly 17.5 trillion possible sequences of the four possible nucleotides. This is a much smaller combinatorial space to reason about compared to plasmid sequences, which are often 5,000bp (roughly $1.995 \times 10^{3,010}$ possible sequences) or greater. Consider the size of the human genome, which is 3,088,286,401bp (roughly $6.373 \times 10^{185,933,683}$ possible sequences).

Other recent research has suggested low-frequency gene mutations are key for predicting or detecting specific types of cancer [144]. The importance of microRNA strands and low-frequency gene mutations with respect to larger biological systems are similar to the importance of patterns, anti-patterns, and fault localization in software testing of highly configurable systems. Mozilla Firefox, for example, has well over $2^{2,000}$ possible configurations available for users [198]. The intuition is that harmful biological DNA “code” such as toxins or cancers could be analogous to configuration and interaction faults in software.

The most influential related work applies supervised machine learning to known toxin and non-toxin DNA sequences, with the goal of being able to predict a priori whether any given sequence is pathogenic. These techniques rely on classification of data points using SVMs and NLP to reason about protein sequences with known characteristics for training sets. The most comprehensive open-source database of protein sequences is UniProt, which is actively maintained [113].

The challenge of applying NLP to DNA sequences is that they are not easily tokenizable and do not have an apparent set of parsable grammar rules for determining syntax or meaning. One common way to reason about a large DNA sequence is to decompose it into shorter sequences, variously described as “motifs,” “n-grams,” or “k-mers” (where “n” and “k” are the base-pair lengths of the subsequences).

Protein sequences from online databases are most-commonly stored in FASTA files which are simply text files with a header describing what is known about the sequence. The sequences are usually encoded in the 20-letter peptide alphabet, where each letter represents an amino acid triplet of base pairs. An example FASTA file for a spider toxin is shown in Figure 6.2.



```
>as:U24-ctenitoxin-Pn1a|sp:P84032 Toxin from venom of the spider Phoneutria
nigriventer with unknown molecular target
ARPKSDCEKHRESTEKTGTIMKLIPKCKENSDYEELQCYEDSKFCVCYDKKGHAASPIST
KVKECGCYLKQKERKDSGRESAIIPQCEEDGKWAKKQLWEFNKSCWCVDEKGEQVGKIHH
DCDSLKCE
```

Figure 6.2: An example FASTA file for a spider toxin encoded in the 20-letter peptide alphabet.

One method of predicting protein characteristics is to split the full sequences into peptide 3-grams (which would be translated into motifs of nucleotide 9-grams), then using frequencies of those motifs to train an NLP system based on the known characteristics of those proteins [9]. Another approach is to combine several machine-learning classifiers and aggregate their predictions (e.g., “1” is toxic and “0” is nontoxic). Assuming each classifier deserves equal weight, one research group combined nine classifiers for a toxin prediction summation score ranging from 0-9 [77].

Other approaches leverage speech-processing techniques by analyzing n-grams of various lengths along with “skip-grams” which ignore certain substrings [107]. The purpose of ignoring skip-grams is to reduce the configuration space to a more manageable size, with an implicit assumption that some substrings are not relevant for protein classification.

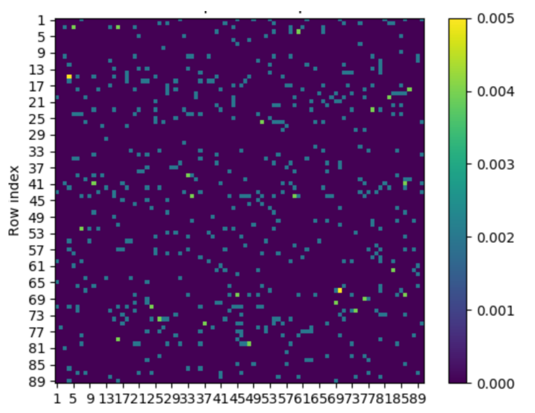
Even early text-compression algorithms have inspired new ways to find patterns as input for training data. Altering the original byte-pair compression algorithm, which was introduced in 1994 [78], led to the concept of "nucleotide-pair encoding" to build data sets for training on phenotype prediction, biomarker detection, and taxonomic analysis [10]. Somewhat similar follow-on work was done for "motif mining" using "peptide-pair encoding" which generates a set of the most-common amino acid pairs from the 20-letter alphabet (20^2, or 400 possible 2-gram sequences), which are then analyzed with a skip-gram neural network [8].

Pse-in-One 2.0[1] is an online toolset which analyzes DNA or peptide sequences to provide statistics and some visualizations which could be used to train support vector machines (SVMs) for various types of predictions [139]. However, their tools appear to only analyze up to 3-mers for peptides (20^3) and 6-mers for nucleotides (4^6) [279]. Figure 6.3a shows Pse-in-One's visualizion of peptide 3-mers of a spider toxin, and Figure 6.3b shows the visualization of nucleotide 6-mers for that same toxin.

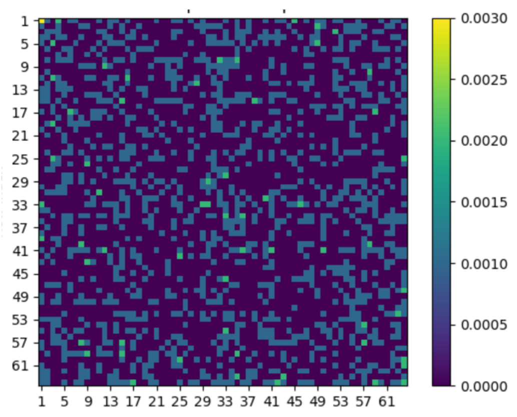
Note that neither visualizations provide meaningful labels for which motifs they represent. It would be more useful for the labels to include the actual characters of the motifs for each row and column, such as "0:AAA" and "63:TTT." Figure 6.3a is especially confusing because it uses a 90x90 matrix for the 8,000 possible peptide 3-mers, resulting in 8,100 data points. That means there are 100 superfluous points, and it is unclear how the rows and columns translate to the actual peptide motifs. Pse-in-One only offers this one type of visualization, sometimes called a two-dimensional heat map.

This work shows it is possible to visualize the motifs in a multitude of other,

[1]The name Pse-in-One was developed from "PseDAC-General," an abbreviation for "pseudo deoxyribonucleic acid compositions for DNA sequences."



(a) Visualization for a spider toxin using a 3-mer of peptides.



(b) Visualization for a spider toxin using a 6-mer of nucleotides.

Figure 6.3: Visualizations for peptides and nucleotides from Pse-in-One [279].

more useful ways, and also to analyze motifs containing a greater number of characters. Analysis can be extended to 6-mers and 8-mers for peptides, and up to 16 nucleotides before running into memory constraints. There are 20^8 $(25,600,000,000)$ possible 8-mers for peptides, and 4^{16} $(4,294,967,296)$ possible 16-mers for nucleotides.

Others have suggested that over-reliance on sequence similarity for protein classification can be insufficient and misleading for a wide variety of proteins, and additional layers of information, such as protein structure, can improve accuracy [161]. There is growing agreement that a multifaceted approach is needed for protein classification [55]. One recent study looked at several factors such as amino acid composition, protein length, molecular weight, hydrophobicity, and molecular charge [169].

6.3 The AlphaFreque Tool

The contribution of this chapter is a tool for counting alphabetical frequencies (AlphaFreque) of longer subsequence motifs of peptide or nucleotides. AlphaFreque extends other work with the following contributions:

Protein Set	Description	Count	Bytes	Source
Araneae toxins	Spiders	1,833	131,928	[95]
ATDB toxins	Venomous animals	3,820	275,701	[93]
BTXPred toxins	Bacteria	183	105,707	[185]
Conotoxins	Predatory sea-snails	6,251	207,350	[114]
DBETH toxins	Bacterial exotoxins	227	91,595	[37]
E. coli toxins	*E. coli*	36	3,140	[7]
NTXPred toxins	Predatory neurotoxins	930	60,791	[186]
BTXPred non-toxins	Bacterial non-toxins	498	164,828	[185]
Human non-toxins	Humans	399	8,864	[7]
NTXPred non-toxins	Predatory non-toxins	370	127,520	[186]
Totals		14,547	1,177,424	

Table 6.1: Statistics of the ten different protein sets used as input.

- Increasing the length of peptide sequence analysis to eight characters (20-letter alphabet);
- Increasing the length of nucleotide sequence analysis to 16 characters (four-letter alphabet); and
- Improving visualizations of sequence frequencies through clearer labeling and translation of data into more meaningful matrix compositions.

6.3.1 Tool Repository

The code for the tool is available in an online Git repository.[2] Also included are the ten sets of input files and two separate aggregate files, one for all toxins and one for all non-toxins. The code base can perform peptide analysis for 6-mers or 8-mers, and nucleotide analysis for 8-mers or 16-mers. It has the option to perform codon optimization for humans, E. coli, or random. For visualizations, there is also a MATLAB script showing how to produce images shown in this chapter based on the output files.

[2] `https://github.com/irrumator88/AlphaFreque`

6.3.2 Peptide Encoding

One insight that makes this work unique is that it is more efficient to count frequencies of a 20-letter alphabet using a base-20 numbering system. This way, a 6-mer can be counted in a matrix of all possible 3-mer rows and all possible 3-mer columns. Similarly, an 8-mer can be counted in a matrix of all possible 4-mer rows and columns. There is thus a one-to-one ratio of each matrix cell to each possible 6-mer or 8-mer. Recall Figure 6.3a from the Pse-in-One tool which used a 90x90 (8,100) matrix for counting all possible 3-mers (8,000).

Given the 20-letter peptide alphabet, each character represents 0-19 in base 20 as shown in Table 6.2. For a 6-mer analysis, there are 20^6 $(64,000,000)$ possible sequences, requiring an 8,000x8,000 matrix. For an 8-mer analysis, there are 20^8 $(25,600,000,000)$ possible sequences, requiring a 160,000x160,000 matrix. The first half of each subsequence is the row, and the second half is the column.

Example calculations are shown in Tables 6.3 and 6.4. At the extremes, the 6-mer AAAAAA is located at [0,0] and VVVVVV is located at [7999,7999]. Somewhere in between those two extremes is NDCMFP, which is located at [864,5074]. Similar calculations are performed for 8-mers, with AAAAAAAA located at [0,0], VVVVVVVV located at [159999,159999], and NDCRMFPR located at [17281,101481]. With this encoding system in base 20, it is possible to progressively scan any FASTA sequence and produce counts of all possible 6-mers or 8-mers.

The basic process for 6-mer counting is described in Algorithm 1. First, instantiate an 8,000x8,000 matrix with initialized values of zero for all cells. Then, progressively scan every six characters of the input file, chopping the left three characters for the row and the right three characters for the column. Translate each pair of the three characters into decimal numbers using the base-20 coding

A	R	N	D	C	E	Q	G	H	I
0	1	2	3	4	5	6	7	8	9
L	K	M	F	P	S	T	W	Y	V
10	11	12	13	14	15	16	17	18	19

Table 6.2: Base-20 encodings for the peptide alphabet.

i[0]	i[1]	i[2]	Sum/Row	j[0]	j[1]	j[2]	Sum/Column
A	A	A		A	A	A	
0x400	0x20	0	0	0x400	0x20	0	0
V	V	V		V	V	V	
19x400	19x20	19	7999	19x400	19x20	19	7999
N	D	C		M	F	P	
2x400	3x20	4	864	12x400	13x20	14	5074

Table 6.3: Example base-20 calculations for 6-mers.

i[0]	i[1]	i[2]	i[3]	Sum/Row	j[0]	j[1]	j[2]	j[3]	Sum/Column
A	A	A	A		A	A	A	A	
0x8000	0x400	0x20	0	0	0x8000	0x400	0x20	0	0
V	V	V	V		V	V	V	V	
19x8000	19x400	19x20	19	159999	19x8000	19x400	19x20	19	159999
N	D	C	R		M	F	P	R	
2x8000	3x400	4x20	1	17281	12x8000	13x400	14x20	1	101481

Table 6.4: Example base-20 calculations for 8-mers.

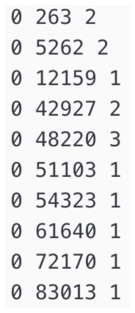
system from Table 6.2, and increment by one the cell associated at that row and column. The process is only slightly different for 8-mers, requiring an exponentially larger matrix.

Algorithm 1 Frequency counting algorithm for 6-mer peptides.

```
1: Input: a concatenated file of peptide sequences
2: instantiate matrix6mer[8000][8000] with zeros
3: for all peptide sequences do scan progressively every six characters
4:     translate left half and right half of 6-mer using base-20 encoding
5:     left half → i; right half → j
6:     matrix6mer[i][j]++
7: end for
8: Output: a 8000x8000 matrix with counts of the input file
```

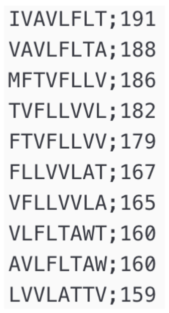
After analyzing the input, AlphaFreque writes two different output files for further analysis. Figure 6.4a shows the first ten rows of output for the first file, which has three space-separated columns. The first column is the X coordinate, the second column is the Y coordinate, and the third column is the frequency of the count of the 8-mer at that X-Y coordinate. Any X-Y coordinate pairs that are not part of the output indicates that there was a count of zero (those 8-mers did not appear) in the input file. The order of the output is sorted first by row-major ascending, then secondarily by column-major ascending.

Figure 6.4b shows the first ten rows of output for the second file, which has two columns separated by a semicolon. Each row is the string representation of the 8-mer followed by its frequency from the input file. The output is sorted descending by frequency. If any 8-mer strings do not appear in the output file, that means they did not appear in the input file.



```
0 263 2
0 5262 2
0 12159 1
0 42927 2
0 48220 3
0 51103 1
0 54323 1
0 61640 1
0 72170 1
0 83013 1
```

(a) First of two sample output files. The three columns represent the X coordinate, the Y coordinate, and the frequency of the 8-mer at those X-Y coordinates.



```
IVAVLFLT;191
VAVLFLTA;188
MFTVFLLV;186
TVFLLVVL;182
FTVFLLVV;179
FLLVVLAT;167
VFLLVVLA;165
VLFLTAWT;160
AVLFLTAW;160
LVVLATTV;159
```

(b) Second of two sample output files. The semicolon-separated values are the 8-mer string and its frequency, sorted descending by frequency.

Figure 6.4: Sample output files from AlphaFreque.

6.3.3 Nucleotide Encoding

All of the chosen protein sequences were originally encoded using the 20-letter peptide alphabet shown in Table 6.2. However, when discovered in nature, they would have been sequenced using the four-letter nucleotide alphabet from actual DNA strands. There are 4^3 (64) possible ways to make a nucleotide triplet from [A, C, G, T/U],[3] so the 20-letter peptide alphabet represents a many-to-one mapping shown in Figure 6.5. For example, there are six different nucleotide triplets that encode for the peptide serine (S): 1) AGC; 2) AGU; 3) UCA; 4) UCC; 5) UCG; and 6) UCU.

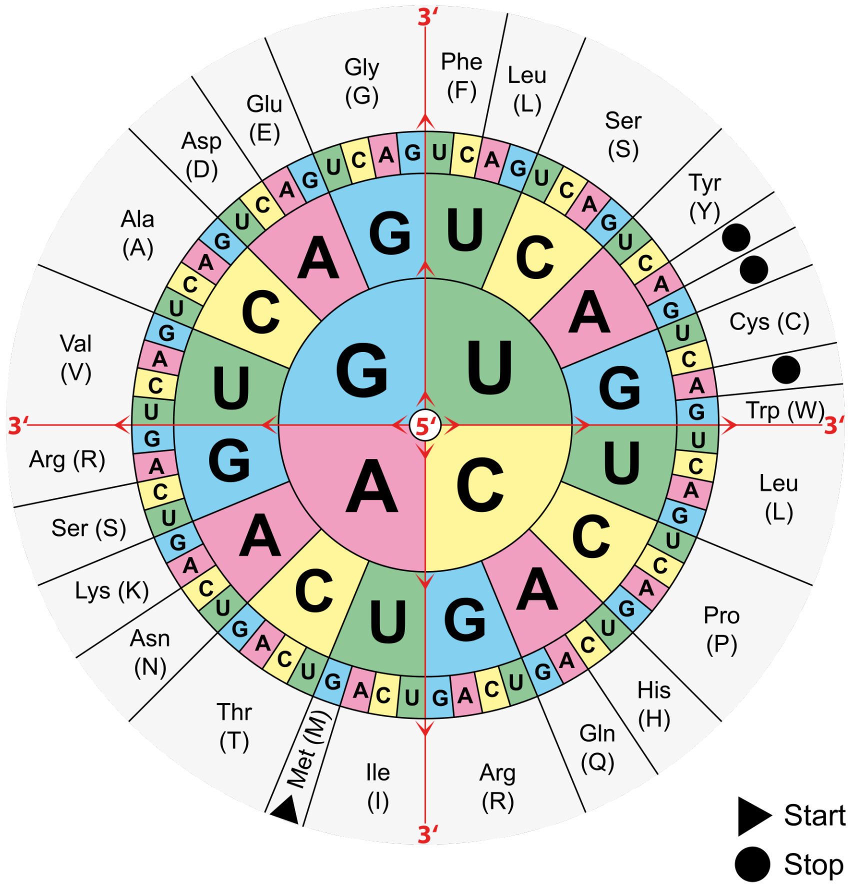


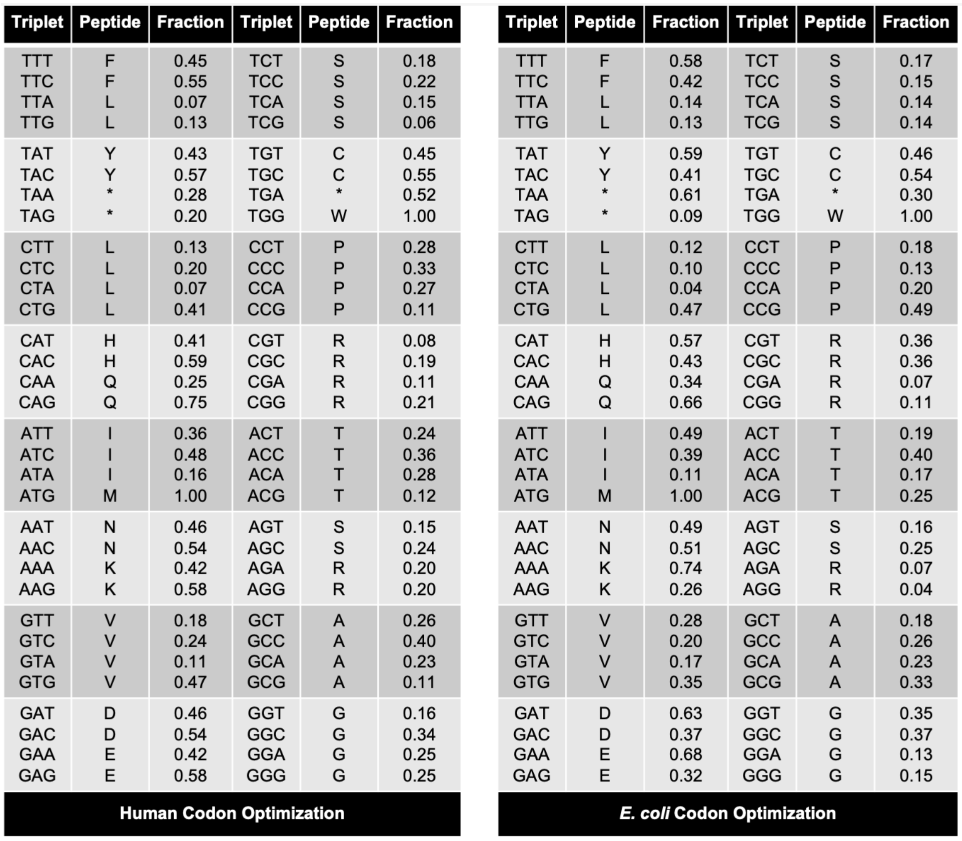
Figure 6.5: A wheel with all 64 nucleotide triplets as a many-to-one mapping into the 20-letter peptide alphabet [283].

AlphaFreque can map peptide sequences into two different sets of DNA nucleotide sequences: one codon-optimized for *E. coli* hosts, and another codon-optimized for human hosts. Codon optimization is based on the most frequently observed mappings for the host species. The rationale for codon optimization is that protein expression can be higher or lower in a host depending

[3]When translating, RNA uses the nucleotide uracil (U) instead of thymine (T).

upon which of the many-to-one peptide mappings are chosen when translating the protein sequence into a DNA nucleotide sequence [29]. AlphaFreque also supports random codon optimization.[4]

Figure 6.6 shows two codon optimization tables, one for humans and one for *E. coli*. As an example of how optimization could differ across species, compare preferences for expressing phenylalanine (F). Humans prefer TTC (55%) over TTT (45%), but *E. coli* prefers TTT (58%) over TTC (42%).



Triplet	Peptide	Fraction	Triplet	Peptide	Fraction
TTT	F	0.45	TCT	S	0.18
TTC	F	0.55	TCC	S	0.22
TTA	L	0.07	TCA	S	0.15
TTG	L	0.13	TCG	S	0.06
TAT	Y	0.43	TGT	C	0.45
TAC	Y	0.57	TGC	C	0.55
TAA	*	0.28	TGA	*	0.52
TAG	*	0.20	TGG	W	1.00
CTT	L	0.13	CCT	P	0.28
CTC	L	0.20	CCC	P	0.33
CTA	L	0.07	CCA	P	0.27
CTG	L	0.41	CCG	P	0.11
CAT	H	0.41	CGT	R	0.08
CAC	H	0.59	CGC	R	0.19
CAA	Q	0.25	CGA	R	0.11
CAG	Q	0.75	CGG	R	0.21
ATT	I	0.36	ACT	T	0.24
ATC	I	0.48	ACC	T	0.36
ATA	I	0.16	ACA	T	0.28
ATG	M	1.00	ACG	T	0.12
AAT	N	0.46	AGT	S	0.15
AAC	N	0.54	AGC	S	0.24
AAA	K	0.42	AGA	R	0.20
AAG	K	0.58	AGG	R	0.20
GTT	V	0.18	GCT	A	0.26
GTC	V	0.24	GCC	A	0.40
GTA	V	0.11	GCA	A	0.23
GTG	V	0.47	GCG	A	0.11
GAT	D	0.46	GGT	G	0.16
GAC	D	0.54	GGC	G	0.34
GAA	E	0.42	GGA	G	0.25
GAG	E	0.58	GGG	G	0.25

Human Codon Optimization

Triplet	Peptide	Fraction	Triplet	Peptide	Fraction
TTT	F	0.58	TCT	S	0.17
TTC	F	0.42	TCC	S	0.15
TTA	L	0.14	TCA	S	0.14
TTG	L	0.13	TCG	S	0.14
TAT	Y	0.59	TGT	C	0.46
TAC	Y	0.41	TGC	C	0.54
TAA	*	0.61	TGA	*	0.30
TAG	*	0.09	TGG	W	1.00
CTT	L	0.12	CCT	P	0.18
CTC	L	0.10	CCC	P	0.13
CTA	L	0.04	CCA	P	0.20
CTG	L	0.47	CCG	P	0.49
CAT	H	0.57	CGT	R	0.36
CAC	H	0.43	CGC	R	0.36
CAA	Q	0.34	CGA	R	0.07
CAG	Q	0.66	CGG	R	0.11
ATT	I	0.49	ACT	T	0.19
ATC	I	0.39	ACC	T	0.40
ATA	I	0.11	ACA	T	0.17
ATG	M	1.00	ACG	T	0.25
AAT	N	0.49	AGT	S	0.16
AAC	N	0.51	AGC	S	0.25
AAA	K	0.74	AGA	R	0.07
AAG	K	0.26	AGG	R	0.04
GTT	V	0.28	GCT	A	0.18
GTC	V	0.20	GCC	A	0.26
GTA	V	0.17	GCA	A	0.23
GTG	V	0.35	GCG	A	0.33
GAT	D	0.63	GGT	G	0.35
GAC	D	0.37	GGC	G	0.37
GAA	E	0.68	GGA	G	0.13
GAG	E	0.32	GGG	G	0.15

***E. coli* Codon Optimization**

Figure 6.6: Codon optimization tables for humans and *E. coli*.

Optimized nucleotide sequences are then translated into two-digit binary: A=00; C=01; G=10; T=11. AlphaFreque's nucleotide encoding is not unique, as it is common (and necessary) to use two bits to encode four characters to avoid

[4]Pse-in-One does not offer any type of codon optimization.

loss of information.[5] Algorithm 2 describes te byte-pair counting based on those binary representations of the protein sequences.

Algorithm 2 Byte-pair counting algorithm for nucleotide-encoded inputs.

```
1: Input: a concatenated file of nucleotide sequences
2: for all nucleotide sequences do convert nucleotide bases into two-bit encoding
3: end for
4: instantiate byte-pair matrix[256][256] with zeros
5: for all bit-encoded sequences do scan every 16 bits progressively and convert them into decimal for a
   left byte i and right byte j
6:     matrix[i][j]++
7: end for
8: Output: a 256x256 matrix with byte-pair counts of the input file
```

The 256x256 matrix stores counts for all possible byte-pairs. For example, cell [0][0] represents the byte pair [00000000][00000000] (the 8-gram AAAAAAAA), and cell [255][255] represents byte pair [11111111][11111111] (the 8-gram TTTTTTTT). As an example somewhere in between those two extremes, cell [121][15] represents byte pair [01111001][00001111] (the 8-gram CTGCAATT).

The sequences were scanned progressively with a sliding frame of 16 bits, moving the frame one character at a time. That is, for a 16-gram of AATTAATTACGTACGT mapping to indices 0-15, the algorithm scanned indices 0-7 (AATTAATT), then indices 1-8 (ATTAATTA), continuing until in the same manner until reaching indices 8-15 (ACGTACGT).

6.4 Case Study

This chapter asks two research questions:

- RQ1: Is it possible to analyze longer sequence motifs compared to the state of the art?

[5]According to Claude Shannon's fundamental paper on information theory [192], the number of bits needed to convey information from an alphabet is the binary logarithm of the number of characters. For nucleotides, $\log_2 4 = 2$ bits. For peptides, $\log_2 20 =\sim 4.322$ bits.

- RQ2: If RQ1 is answered yes, is it possible to provide more meaningful visualizations for those motifs?

6.4.1 Study Methods

For input files, ten different sets of protein sequences were manually selected that had been curated and clearly described as either toxins or non-toxins in their FASTA headers. The total dataset comprised 14,547 proteins from the categories shown in Table 6.1. The proteins were analyzed in two different formats: 1) as the original peptide encodings; and 2) mapped into DNA nucleotides. Pse-in-One was used as a benchmark tool for comparison against AlphaFreque.

For the 6-mer peptide and 8-mer nucleotide calculations, we ran all experiments on the same laptop with a 2.5GHz Quad-Core Intel Core i7 with 16GB of RAM. For the 8-mer peptide and 16-mer nucleotide calculations, we ran all experiments on the same computing cluster with 7232 Intel Xeon cores with a maximum Java memory pool of 500GB, reserving 480GB for the heap. The average cluster runtime for 16-mer nucleotides was 6 minutes and 24 seconds, and the average cluster runtime for 8-mer peptides was 9 minutes and 48 seconds.

The core metric for the study is the maximum integer (k) for which we could successfully determine accurate counts of all possible combinations of k-length subsequences appearing in the input files. There are two types of input files: 1) nucleotide encodings using a four-letter alphabet; and 2) peptide encodings using a 20-letter alphabet.

6.4.2 Results: RQ1

The tool was run using the ten input sets listed in Table 6.1. For peptide encoding, all ten sets were scanned for 6-mers and 8-mers. For nucleotide encoding, all ten

Input File	Peptide 6-mers found	Ratio (to 20^6)	Peptide 8-mers found	Ratio (to 20^8)
Araneae toxins	53,004	0.0828%	58,853	0.000230%
ATDB toxins	120,697	0.1886%	141,264	0.000552%
BTXPred toxins	88,029	0.1375%	93,932	0.000367%
Conotoxins	62,164	0.0971%	73,633	0.000288%
DBETH toxins	102,746	0.1605%	106,071	0.000414%
E. coli toxins	3,828	0.0060%	3,852	0.000015%
NTXPred toxins	43,669	0.0683%	48,050	0.000188%
BTXPred non-toxins	136,758	0.2137%	144,943	0.000566%
Human non-toxins	7,628	0.0119%	7,687	0.00003%
NTXPred non-toxins	140,668	0.2197%	146,410	0.000571%
All toxins	368,977	0.5765%	414,132	0.001618%
All non-toxins	164,087	0.2564%	173,275	0.000677%

Table 6.5: Number of possible k-mers discovered in each input file, showing relative sparseness of output matrices.

sets were first optimized for human and *E. coli* hosts, and then scanned for 8-mers and 16-mers. In addition, two aggregate sets were used as input, one of all toxins and another of all non-toxins.

AlphaFreque successfully analyzed all input files using 6-mers and 8-mers for peptides, and 8-mers and 16-mers for nucleotides. An analysis is successful if the output is an accurate count of the frequencies of all possible k-mer combinations appearing in the the input file. Results for peptide encoding are summarized in Table 6.5 which shows how many of the possible k-mer motifs were found in all of the different input files. Percentage ratios are also reported to highlight the relative sparseness of the output matrices. Based on these results, RQ1 is answered in the affirmative.

Top three 6-mer peptide motifs (all toxins)		
Motif	X,Y Coordinates	Frequency
VLFLTA	7813,4320	637
AVLFLT	390,5416	628
VAVLFL	7619,4270	421
Top three 8-mer peptide motifs (all toxins)		
Motif	X,Y Coordinates	Frequency
IVAVLFLT	79619,85416	397
VAVLFLTA	152390,108320	391
MFTVFLLV	101539,108219	376

Table 6.6: Top toxin peptide motifs.

6.5 Results: RQ2

6.5.1 Peptide Encoding

AlphaFreque includes MATLAB code to generate several different types of visualizations from the output data. It can be difficult to interpret large matrices with millions of data points, so the following visualizations only show matrix cells which are three standard deviations above the mean, as these are more likely to be of interest evaluating protein characteristics.[6]

Figures 6.7 through 6.12 show three new types of visualizations for 6-mers for the all toxin and all non-toxin supersets: 2-D scatter plots; 3-D scatter plots; and 3-D stem plots. Recall that the only type of visualization offered by Pse-in-One is a 2-D heat map as seen in Figure 6.3. Figures 6.13 through 6.18 show the same types of visualizations for 8-mers. Unlike Figure 6.3, the axes have meaningful labels with the actual characters of the peptide motifs, and unlike Figure 6.3a, there are no superfluous data points.

For all of the visualizations, the x-axis is the first half of the characters of the motif, the y-axis is the second half of the characters of the motif, and the z-axis is

[6]Data points beyond three standard deviations from the mean are commonly referred to as "outliers." [134]

the frequency of the motif. There are noticeable spikes for certain motifs, which biologists can isolate for further investigation. Table 6.6 shows the top three peptide motifs from the all-toxins input. Given this information and Figure 6.9, a biologist could easily see that the top two motifs visually stand out as the two yellow circles. Any nearby motifs with similar high frequencies are also candidates for further investigation. The spikes are also relatively easy to see in Figure 6.11, but difficult to find in the 2-D scatter plot of Figure 6.7. This could be addressed by different hovering techniques or simply coloring those data points differently.

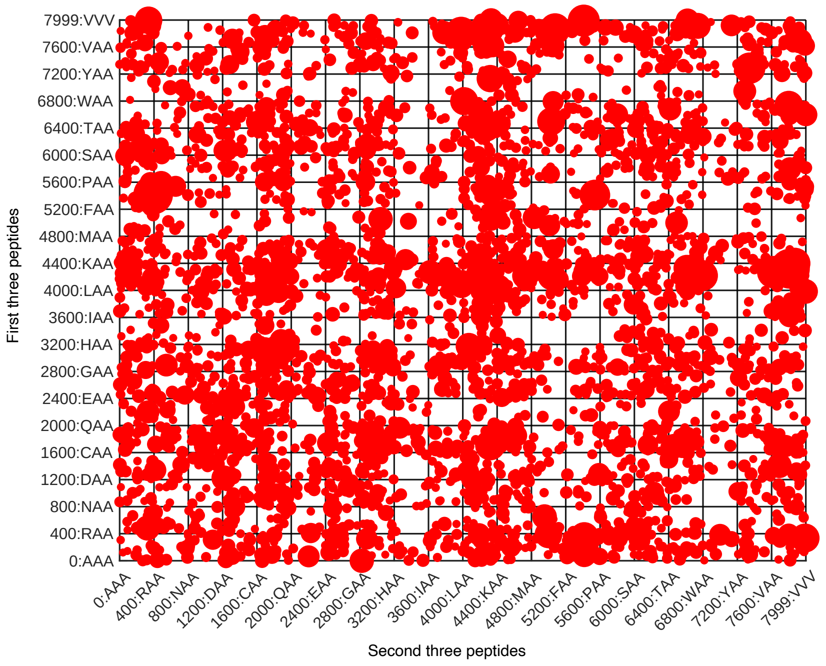


Figure 6.7: A scatter plot of all 6-mer toxin sequences three standard deviations above the mean.

6.5.2 Nucleotide Encoding

It is easier to visualize smaller matrices, and with nucleotide encoding, each possible byte pair for 8-mers can represent exactly one pixel of a 65,636-pixel image. Just as Pse-in-One provides 2-D heat maps, Figure 6.19 shows similar visualizations for 8-mers for all toxins and all non-toxins optimized for human

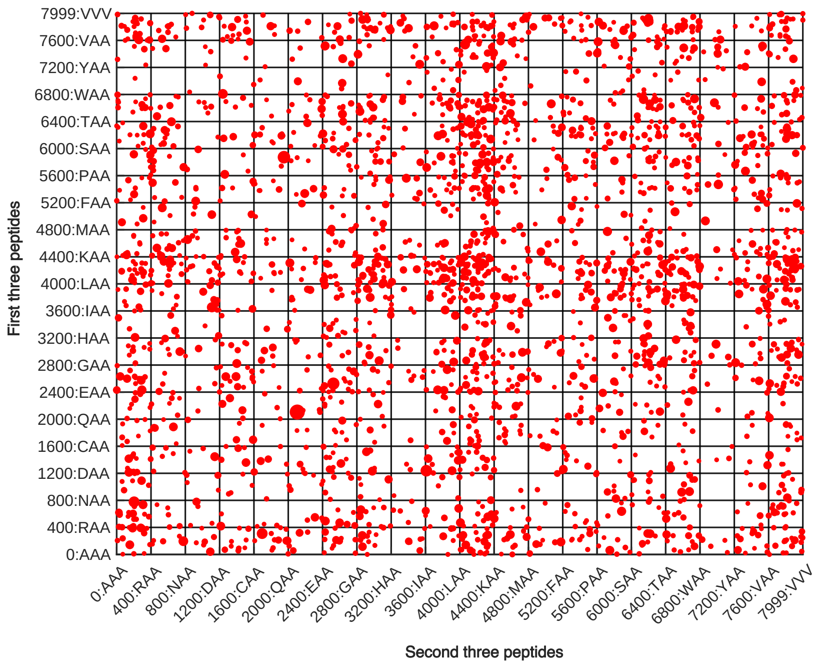


Figure 6.8: A scatter plot of all 6-mer non-toxin sequences three standard deviations above the mean.

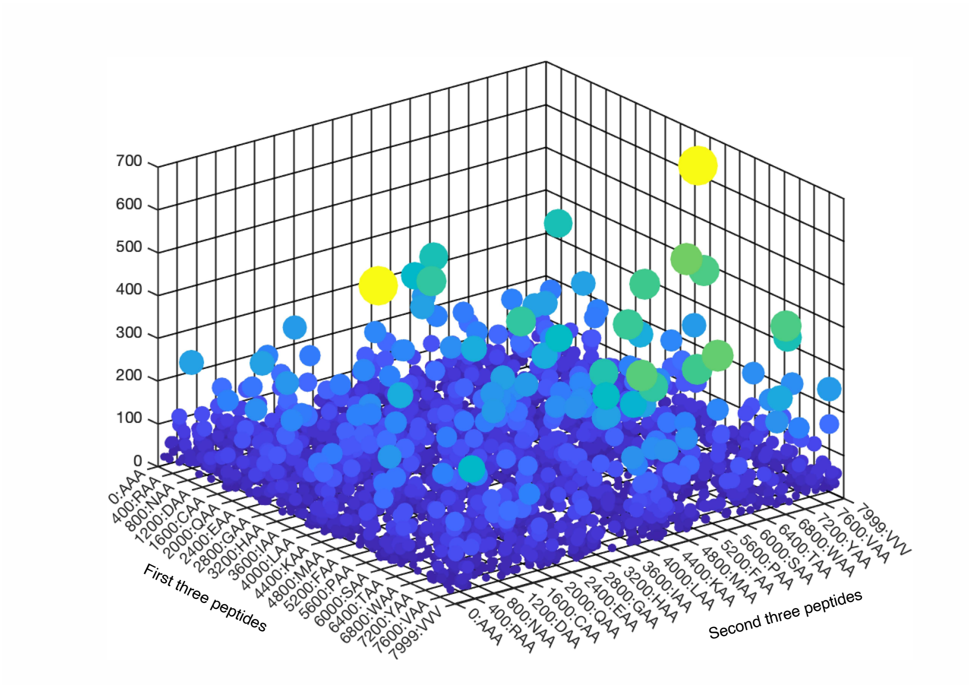


Figure 6.9: A 3-D scatter plot of all 6-mer toxin sequences three standard deviations above the mean.

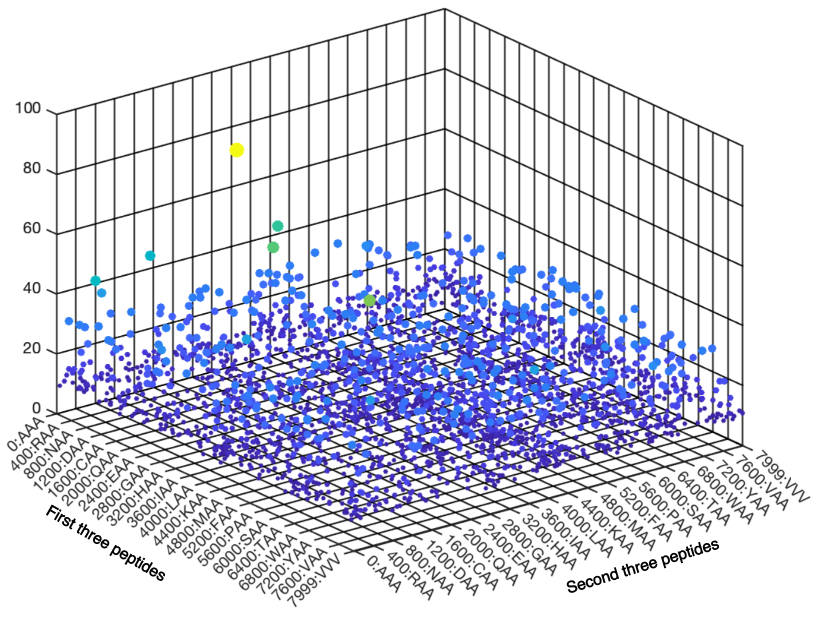


Figure 6.10: A 3-D scatter plot of all 6-mer non-toxin sequences three standard deviations above the mean.

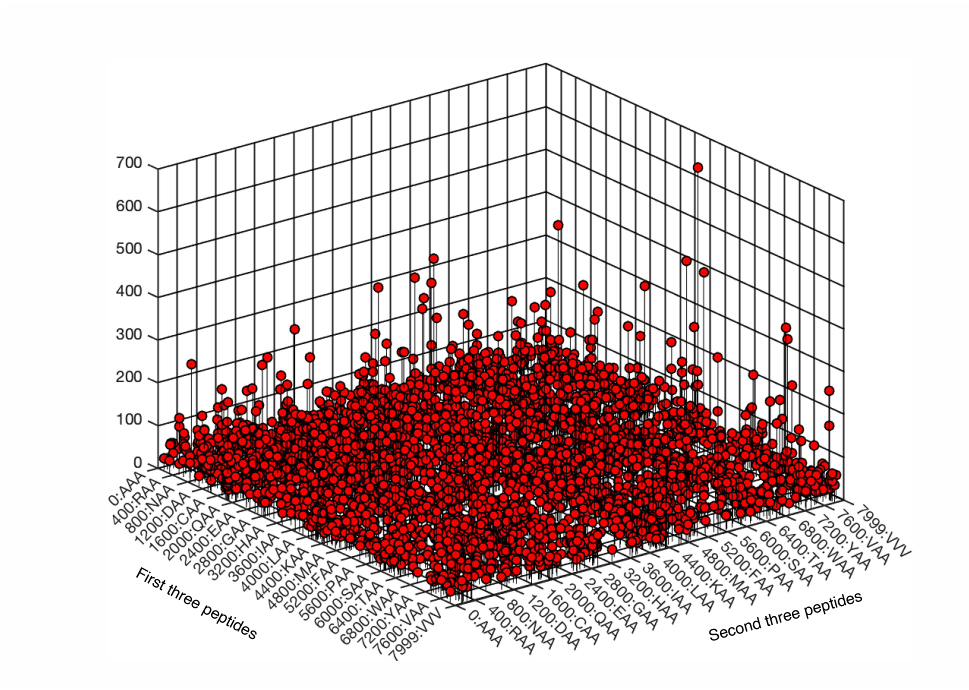


Figure 6.11: A 3-D stem plot of all 6-mer toxin sequences three standard deviations above the mean.

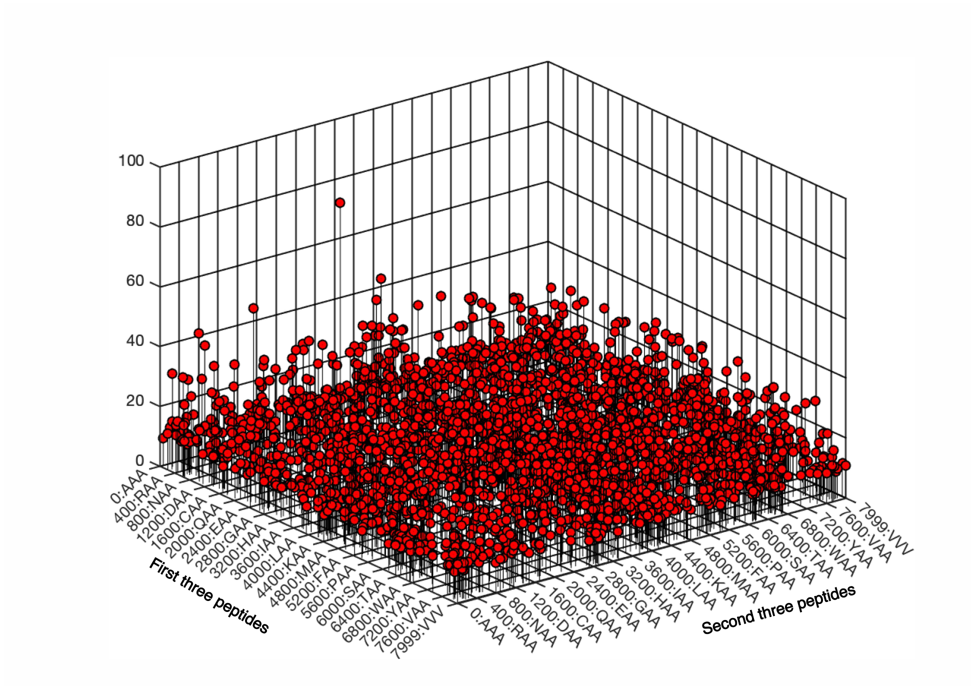


Figure 6.12: A 3-D stem plot of all 6-mer non-toxin sequences three standard deviations above the mean.

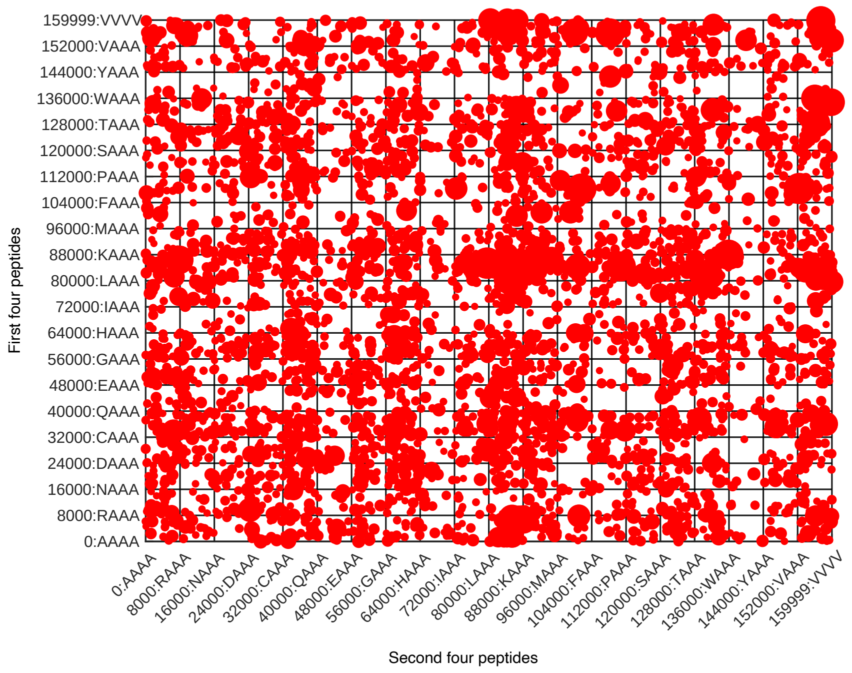


Figure 6.13: A scatter plot of all 8-mer toxin sequences three standard deviations above the mean.

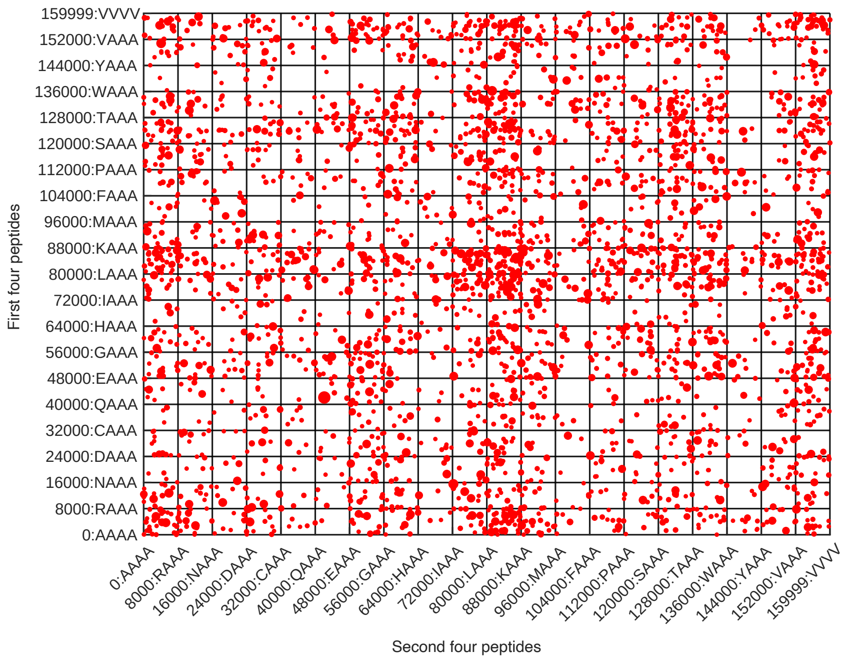


Figure 6.14: A scatter plot of all 8-mer non-toxin sequences three standard deviations above the mean.

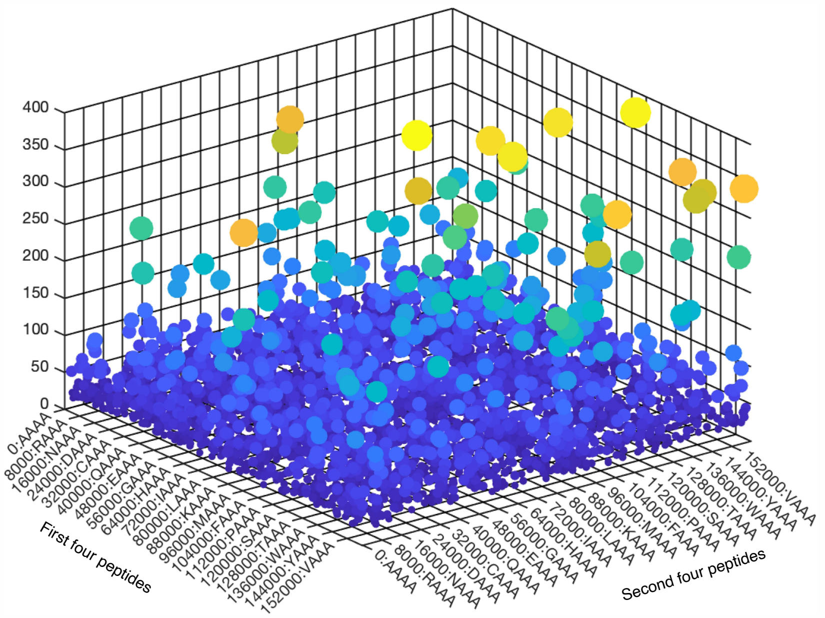


Figure 6.15: A 3-D scatter plot of all 8-mer toxin sequences three standard deviations above the mean.

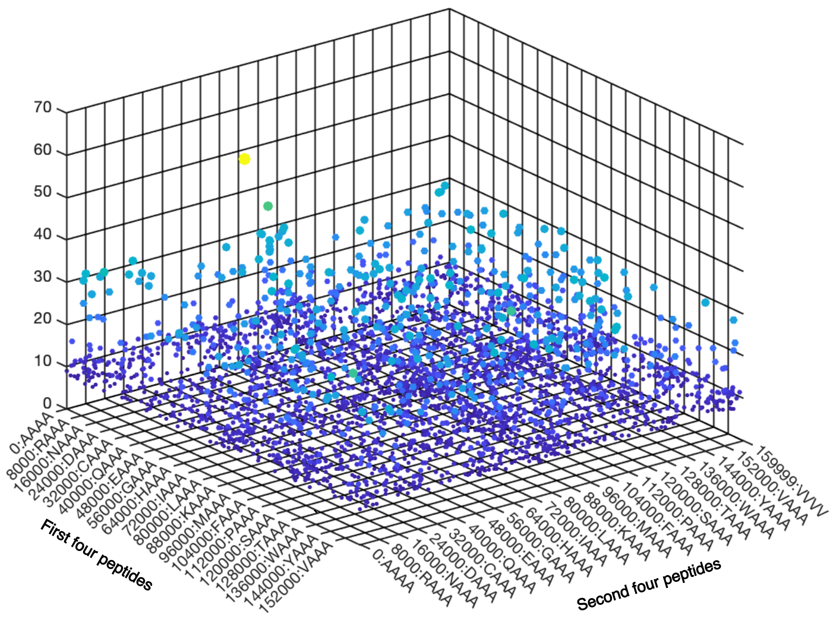


Figure 6.16: A 3-D scatter plot of all 8-mer non-toxin sequences three standard deviations above the mean.

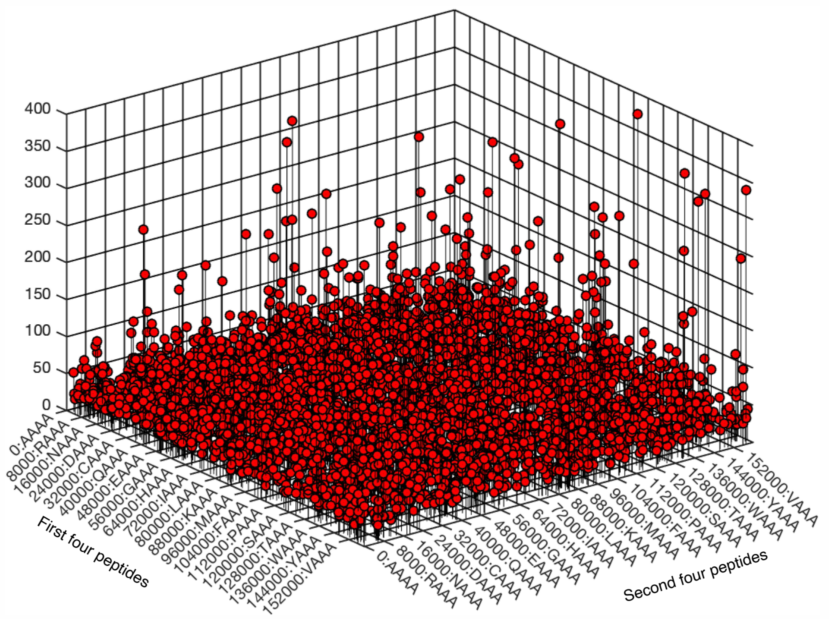


Figure 6.17: A 3-D stem plot of all 8-mer toxin sequences three standard deviations above the mean.

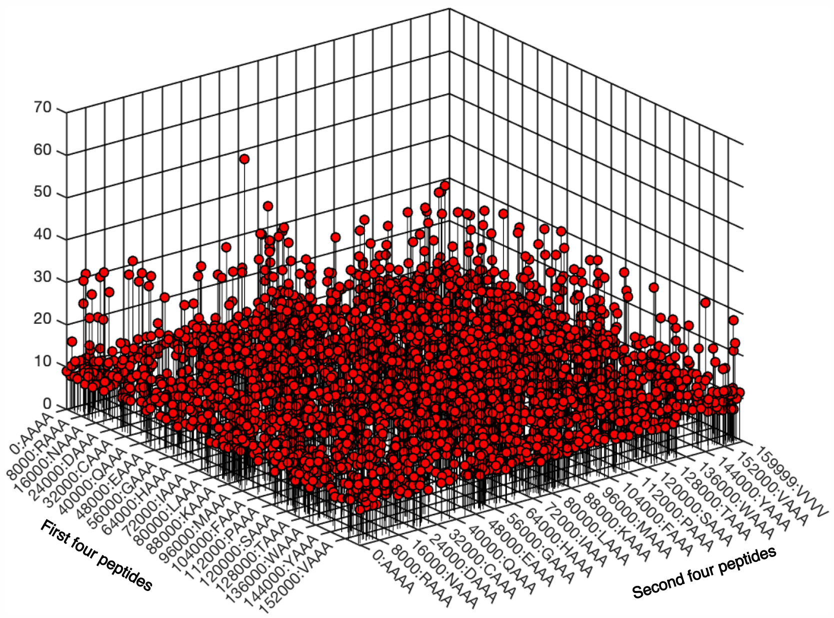
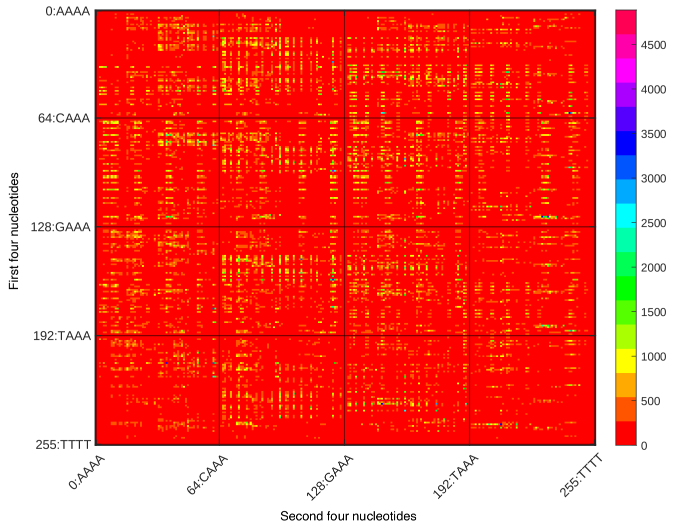


Figure 6.18: A 3-D stem plot of all 8-mer non-toxin sequences three standard deviations above the mean.

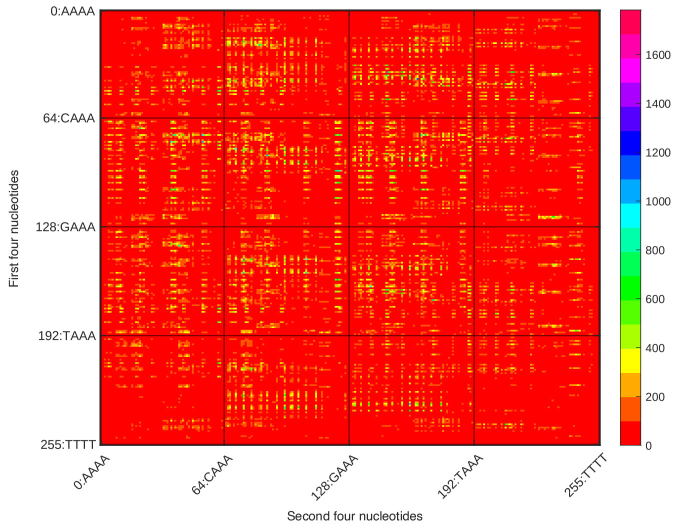
hosts. AlphaFreque also provides MATLAB code to create 3-D heatmaps, as seen in Figure 6.20. Again, the axes have more meaningful labels than those in Figure 6.3b, because they translate into the actual characters of the nucleotide motifs.

To show the differences between human and *E. coli* optimization, Figure 6.21 shows two-dimensional counts for all toxins and all non-toxins optimized for *E. coli* hosts, and Figure 6.22 shows that same data in three dimensions.

AlphaFreque thus offers several new visualizations compared to Pse-in-One, and labels are more meaningful because they translate into the actual motif characters. The images are less confusing because there are no superfluous cells. Results are summarized in Table 6.7 and RQ2 is answered in the affirmative.

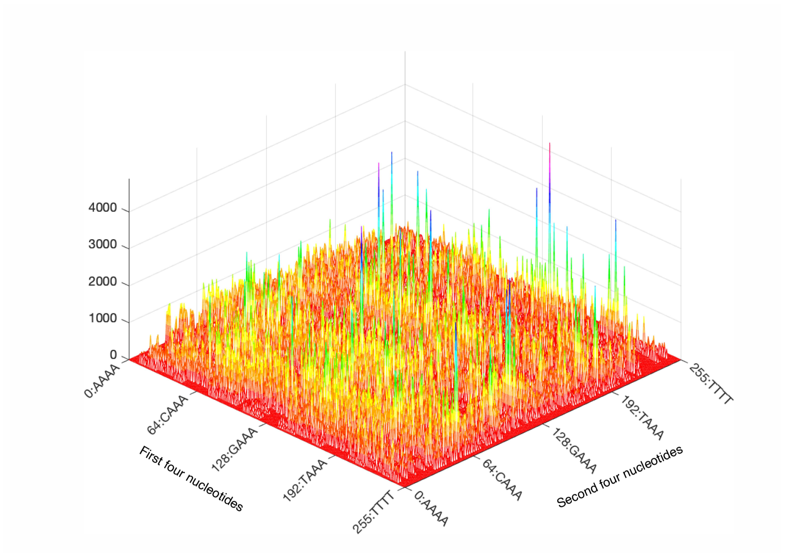


(a) A 2-D heatmap of all toxins using human nucleotide optimization.

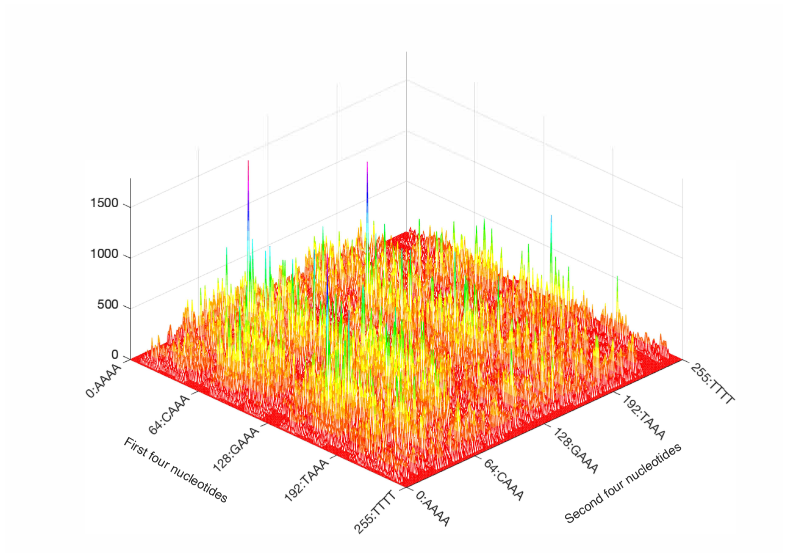


(b) A 2-D heatmap of all non-toxins using human nucleotide optimization.

Figure 6.19: Two-dimensional visualizations of all toxins and all non-toxins (human optimized).

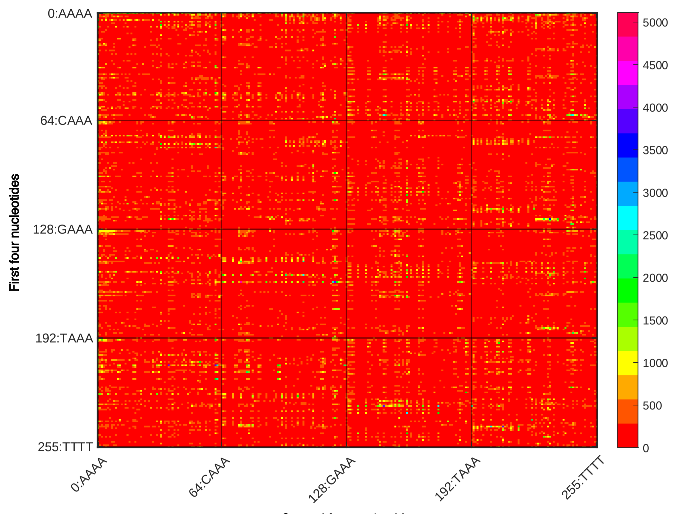


(a) A 3-D heatmap of all toxins using human nucleotide optimization.

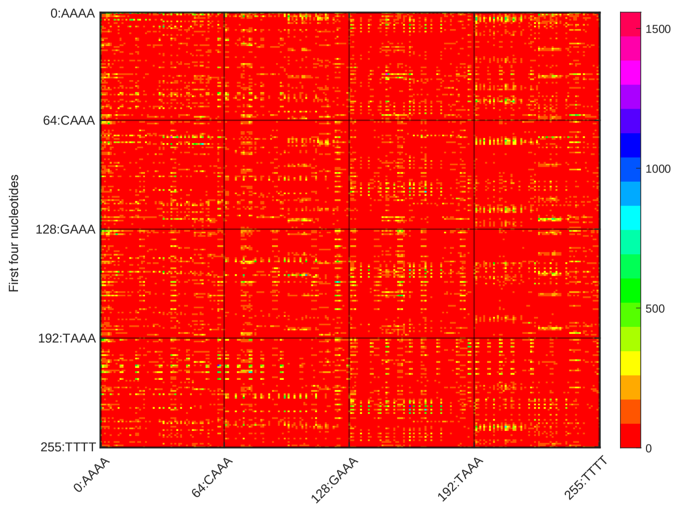


(b) A 3-D heatmap of all non-toxins using human nucleotide optimization.

Figure 6.20: Three-dimensional visualizations of all toxins and all non-toxins (human optimized).

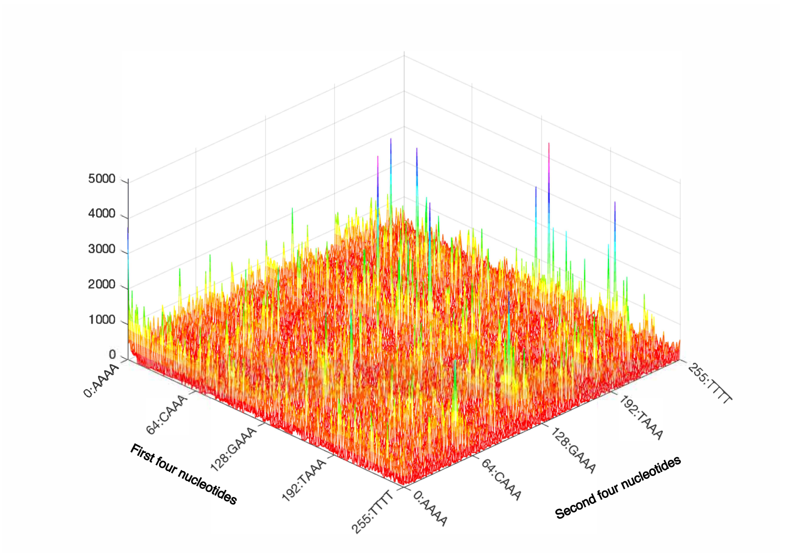


(a) A 2-D heatmap of all toxins using *E. coli* nucleotide optimization.

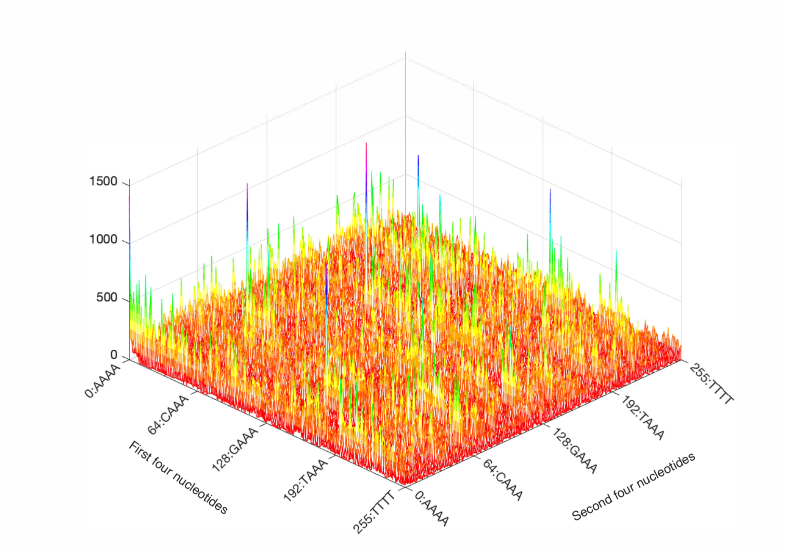


(b) A 2-D heatmap of all non-toxins using *E. coli* nucleotide optimization.

Figure 6.21: Two-dimensional visualizations of all toxins and all non-toxins (*E. coli* optimized).



(a) A 3-D heatmap of all toxins using *E. coli* nucleotide optimization.



(b) A 3-D heatmap of all non-toxins using *E. coli* nucleotide optimization.

Figure 6.22: Three-dimensional visualizations of all toxins and all non-toxins (*E. coli* optimized).

Tool	Visualizations Supported	Are Labels and Cell Numbers Meaningful?
Pse-in-One	•2-D heat map	No
AlphaFreque	•2-D heat map •3-D heat map •2-D scatter plot •3-D scatter plot •3-D stem plot	Yes

Table 6.7: Comparison of k-mers which can be analyzed using Pse-in-One vs. AlphaFreque.

6.6 Discussion

As an initial investigation of the usefulness of the output data for determining whether a protein sequence was a toxin or non-toxin, we randomly took 10% of each of the toxin and non-toxin outputs for the frequencies of 8-mer peptides. The data was used to train models for MATLAB's Classification Learner, but none of the standard settings produced models which were statistically significant for determining whether sequences were toxins or not. It thus remains to determine what methods are best for analyzing the data for toxin classification, a task which we leave for future work.

Regardless, biologists can use AlphaFreque to analyze any nucleotide or peptide inputs, not just those which have been clearly identified as toxins or not. In particular, the 8-mer peptide analysis is useful for studying motifs which are the length of microRNA. Also, biologists might further investigate the data from Table 6.5, which reports the relative sparseness of the output matrices for different sets of protein inputs.

6.7 Conclusion and Future Work

This chapter describes an improved method and tool for analyzing and visualizing motifs in nucleotide and peptide sequences. AlphaFreque can analyze peptide sequences up to 8-mers and nucleotide sequences up to 8-mers. The tool supports the creation of several new types of visualizations which feature meaningful labels and do not include superfluous data.The tool also allows for three types of codon optimization: 1) human; 2) *E. coli*; and 3) random.

It is possible to extend AlphaFreque's codon optimization to other species, assuming the optimization tables are known. Another extension could be developing new visualizations appropriate to use as input for machine-learning classifiers. The output is a large set of xyz coordinates which can be plotted in many ways, but not all of them are easy to interpret, especially for the largest sets generated by peptide 8-mers and nucleotide 16-mers. It would be especially useful to make the tool available through a web-based application, along with additional ways to present the data within visualizations which pop out the frequencies and motifs as users move the mouse over different data points.

Another helpful feature would be to perform comparisons of different output files. Some suggestions include finding the union and intersection among the most frequent or least frequent motifs, as well as other general methods for filtering output.

Lastly, due to the exponential nature of the algorithm, it would be challenging to create a tool that analyzes longer subsequences. However, the resulting matrices are relatively sparse, as shown in Table 6.5. That is, most of the entries are zero because those sequences do not occur in the input files. For example, when scanning for 8-mers in the all-toxins file, only 414,132 of the possible 20^8 8-mers appear, meaning a large amount of memory is allocated but never needed for incrementing counts. Depending upon characteristics of input files, the sparseness of the matrices should grow as the length of subsequences is increased.

Instead of instantiating a matrix full of zeros to start, one major improvement for memory usage would be to only track frequencies of motifs upon their first discovery. Each discovered motif could be a separate object with a counter variable.

Chapter 7

TreAssure: Integrating Safety and Security Artifacts to Build Assurance Cases for Synthetic Biology Applications

7.1 Introduction

Biological software, like traditional software, needs to be tested, validated, and verified for correctness within its run-time environment. Not only do cells perform functions, but they can also interact with other life and the environment, meaning they require high assurance and are safety critical [44, 70]. In addition, there are molecular systems which are being designed to be used in specific safety-critical scenarios such as for use in diagnostic or therapeutic systems [60]. Regulators typically expect some type of certification for safety-critical systems and development processes [124], but it is not clear what types of certification are appropriate for biological software.

As with any technology, it is challenging to verify that system designs are correct and that they will operate as intended in their target environments. For example, SB gene drives have been tested on specific species of mosquitoes to limit transmission of the Zika virus, but it is unclear what the long-term consequences might be for the greater ecosystem [20]. As in other domains, SB applications can fail or result in unintended consequences through mistakes or unknown interactions among components or with the environment. As SB applications become more widespread, there is also growing attention to the potential effects of negligent or malevolent behavior [153]. Therefore, any regulatory framework should require certifications for both safety and security.

In response to the growth of SB applications, there is growing attention for improved regulation [73], but there has been little progress from a national or international perspective to effectively regulate in a way which promotes safety and security while at the same time not stifling innovation [69, 148]. Initial steps are needed to assist stakeholders in generating meaningful artifacts for certifying the safety and security of SB applications. Some initial research has suggested assurance cases can be built for SB as one way to facilitate regulation [44, 70], yet there are several impediments to achieving this goal.

It is challenging to create assurance cases for SEBOs because they are living organisms which can evolve or mutate out of their engineered behavior or escape their intended environments. It is also difficult to effectively monitor and observe the state of engineered bacteria whose populations are typically billions of cells. Finally, synthetic biologists likely lack the experience and background knowledge needed to create assurance cases.

There has been some research supporting assurance for specific aspects of the SB domain, but little research supporting the overall process of generating assurance cases for SB. Examples include ways to assure the robustness of SEBOs [60, 230] and proposing dynamic assurance cases to address biological evolution [44]. There remains a need to build a family of assurance and safety techniques for SB.

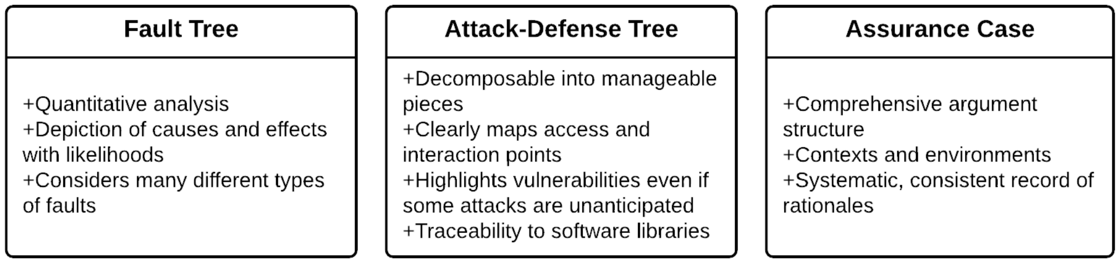
This chapter looks to the safety community and leverages prior work on traceable links between safety and security artifacts of fault trees and attack-defense trees [111, 133]. Those two artifacts are a logical starting point because regulators typically expect them, and engineers already have a strong familiarity with creating them. Importantly, those artifacts naturally provide foundations for safety and security requirements by identifying mitigations, and a regulator should expect to see evidence that directly traces to support implementation of those mitigations. Serendipitously, the leaves of assurance cases are evidence nodes.

Another benefit of assurance cases is they support several types of nodes. Context nodes are especially important for SEBOs because regulators should expect certification that they are safe and secure within an intended environment. Regulators do not expect airplanes [53, 183] or medical devices [30, 137, 223] to operate safely and securely in all possible contexts, and assurance cases explicitly address this.

Fault trees and attack-defense trees could be integrated into a new workflow which helps build the assurance case. The assurance case is an important addition because it allows for an iterative process for identifying assumptions, justifications, contexts, and empirical evidence to show critical hazards and faults from have been properly considered and mitigated. However, these different artifacts are normally created by different people and are independent of each other. There is no single approach to merge all three.

In this work we propose a new approach called “TreAssure” that starts with safety and security artifacts and merges them into an assurance case. The benefit of this approach is that it starts with the simplest diagrams and helps automate creation of the most complex diagram, the assurance case. It provides traceability between the security and safety requirements to the assurance evidence, providing the set of experimental evidence needed by regulators, including a link back to specific requirements. As safety and security requirements are added or changed, the associated assurance case can be flagged for relevant updates.

The next section presents some background and a motivating example.



Fault Tree

+Quantitative analysis
+Depiction of causes and effects with likelihoods
+Considers many different types of faults

Attack-Defense Tree

+Decomposable into manageable pieces
+Clearly maps access and interaction points
+Highlights vulnerabilities even if some attacks are unanticipated
+Traceability to software libraries

Assurance Case

+Comprehensive argument structure
+Contexts and environments
+Systematic, consistent record of rationales

Figure 7.1: Beneficial qualities of the three different artifacts.

Section 7.3 provides an overview of TreAssure, followed by a case study in Section 7.4 and related work in Section 7.6.

7.2 Motivation and Background

This section describes the need for a new approach, provides some background on the individual artifacts used in this paper, and presents a motivating example for TreAssure.

7.2.1 Background

SB applications are engineered, programmed, and safety-critical, naturally lending themselves to assurance cases [44, 70]. However, there is currently no requirement to produce safety or security artifacts for SB applications, and synthetic biologists are unlikely to have experience creating such documents [70]. To encourage effective compliance, proposals for new regulatory frameworks should be based on familiar concepts to encourage collaboration among stakeholders, and there should be guidance regarding how to generate effective artifacts. Safety analysts, security analysts, and regulators are already accustomed to fault trees and attack-defense trees, but it is not obvious how those two artifacts should inform the creation of assurance cases.

7.2.2 Three Different Artifacts

Fault trees, attack-defense trees, and assurance cases all have their own advantages and disadvantages. Figure 7.1 shows the beneficial qualities of the three different artifacts, none of which overlap.

Fault trees provide a quantitative analysis for differing types of faults. They are useful because they provide an easy way to visualize a cause-and-effect model. They consist of fault nodes along with event nodes and allow for probabilities and severity annotations. They use Boolean logic to ensure soundness and are commonly used by regulators. They focus on single component failures, not system failures [133] or independent events which together lead to a fault. The process of identifying the paths to a failure in a fault tree is often a black box and there is little guidance for analysts on how to find those paths because the tree is simply the result of the analysis [133]. Moreover, there is great variability in the quality of fault trees because they depend upon the analyst's expertise in creating and understanding them [133]. Support for fault-tree analysis in critical systems has recently been aided by a proposed SysML profile for fault trees [41].

Attack-defense trees are used to show methods of protection of a system from security attacks. They include both the method of attack and the defense that the system will take to mitigate these attacks. They typically have both attack nodes and mitigation nodes, and are decomposable to provide traceability to system libraries. Attack-defense trees suffer from similar problems as fault trees because they only reflect known attacks, further limited by whether the analyst is actively aware of them and identifies them in the analysis [58].

Assurance cases (sometimes called "safety cases") provide a comprehensive argument structure showing that a system will operate correctly and safely with leaf nodes containing evidence to validate the assurance case claims. They have goal nodes and strategy nodes, which can link to assumptions, justifications, and contexts, which determine the safe operating environment. While assurance cases use the Goal Structuring Notation (GSN), there is a great degree of freedom in how analysts can structure them. That is, two different analysts might create two completely different assurance cases for the same system which are both valid but look nothing like each other and have completely different text within the nodes. This lack of rigidity or formalism can lead to readability issues and a resulting lack of clarity [129]. This problem can be alleviated through the development of well formed patterns [53] and domain-specific sub-patterns which can be more easily instantiated by end users [70].

Assurance cases are also expected to handle a large and wide variety of evidence to support the arguments, which leads to a lack of coherence and consistency [129]. Because assurance cases rely heavily on natural language, which itself offers a great degree of freedom, it can become difficult to correctly, coherently, and sufficiently craft an argument structure with appropriate context, assumption, and justification nodes [129].

All three artifacts are time-intensive and require deep understanding and expertise. In particular, assurance cases have readability and scalability issues. However, each of them have inherent value and if they can be developed as a connected set of artifacts, their usefulness would be greater than the sum of their parts. This could improve the regulatory process for systems which span multiple engineering domains.

7.2.3 Motivating Example

We present a motivating example based on the temperature-sensitive kill switch from the methane-inhibiting application based on the 2017 UNL iGEM team [282] (see Section 7.4 for a more complete description). Stirling, et al., define kill switches as “artificial systems that result in cell death under certain conditions.” [196]

A complete system application would involve adding the engineered bacteria to cattle feed so they become part of the biome in the digestive system where they are able to interfere with the metabolic pathway which produces methane within the cattle rumen [282]. To prevent the engineered bacteria from interacting with the general environment, one safety requirement could be that they are only able to live and operate within the digestive system of the cattle. To ensure the bacteria do not survive after excretion, a temperature-sensitive kill switch (cryodeath) is chosen to meet that safety requirement [196]. As shown in Figure 7.2, the kill switch is stable at temperatures inside the cattle (around 37° C), and activates at temperatures of 22° C and below.

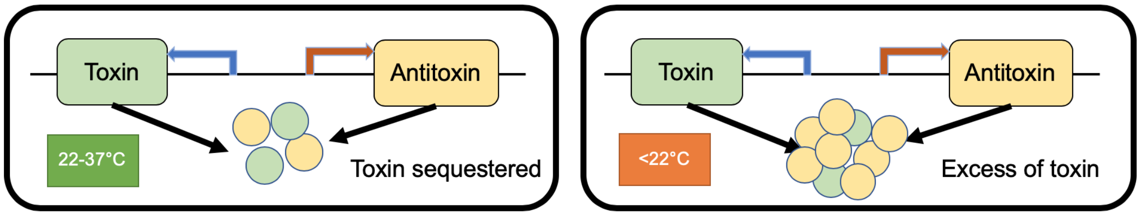


Figure 7.2: Cells produce roughly the same amount of toxins and antitoxins at the permissive temperature range, but an excess amount of toxins at 22° C and below. Image adapted from [196].

It is unclear, from an engineering or regulatory perspective, what types of artifacts would be appropriate for attesting to the safety and security of such a safety-critical system. However, a regulator might ask for a fault tree. From that artifact we would be able to list the failure of the kill switch as a potential leaf node and we would have several paths that could cause this.

This fault-tree is useful, but does not provide empirical evidence or mitigations if the kill switch fails. Instead, we might turn to an attack-defense tree, another artifact a regulator may require. A node in the attack-defense tree might describe an attack by a malicious actor which manipulates the ambient temperature to defeat the kill switch, for which we can provide mitigations such as keeping the cattle and the engineered organisms in a secured, temperature-controlled facility. This view of safety is also important, but again shows only one facet: security.

Johnson and Kelly have already identified the challenges of co-assurance across the domains of safety and security [111]. They introduced the Safety-Security Assurance Framework (SSAF) to help link fault-tree artifacts with attack-tree artifacts, recognizing that a safety expert might not necessarily be a security expert, and vice versa. However, it should be intuitive to both safety and security experts that an attack can lead directly to a failure mode.

While the connection of these two artifacts provides safety and security analysis, neither provides the necessary level of detail and evidence that this application works as expected in its operational environment. An important motivation for implementing kill switches is biological containment and confining an engineered biological system within specific environments [196]. This suggests the importance of adding an assurance case, because its structure contemplates operational contexts and identifies what evidence or experiments are needed to satisfy the arguments indicating each requirement is met. The process of building assurance cases incorporates: 1) domain modeling and analysis; 2) system development and verification; and 3) safety analysis and argument development, the results of which can be organized into tables [50].

Assuming a fault tree and attack-defense tree are properly instantiated, the next phase of work involves cross-domain sharing of knowledge between fault trees and attack-defense trees to semi-automate assurance cases, rather than building them from scratch. This will lead to traceable links between safety and security and point to evidence that demonstrates mitigations are in place. As part of the process, analysts can identify agents responsible for implementing and monitoring the applications. This is the goal of TreAssure, which is present next.

7.3 TreAssure: General Process

TreAssure uses the information in the fault trees and attack-defense trees to formulate an assurance case that addresses both safety and security. While developing the fault tree and attack-defense tree, the safety and security experts will be able to identify critical faults and hazards which can lead to an unsafe state. Each leaf node from the fault tree and attack-defense tree should be traceable to at least one evidence node, a process which could be semi-automated.

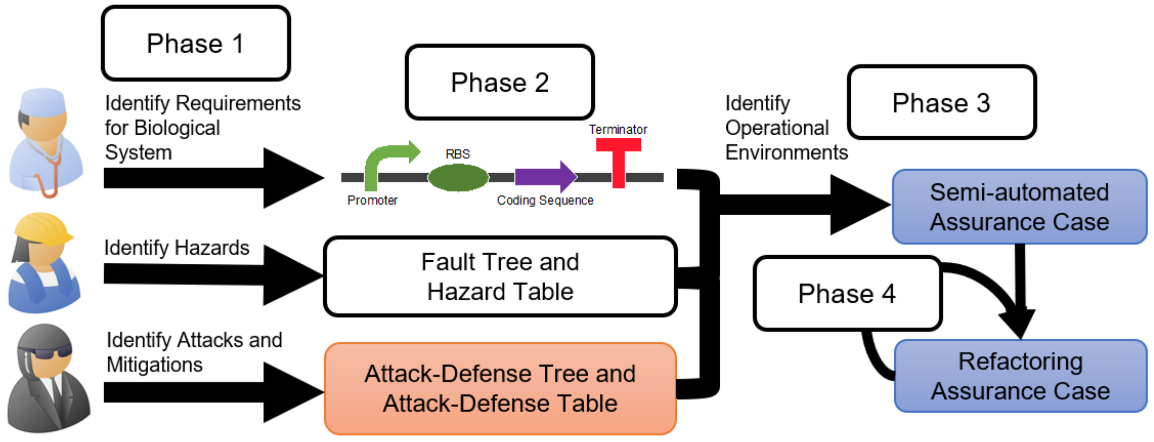

Phase 1
Identify Requirements for Biological System
Identify Hazards
Identify Attacks and Mitigations
Phase 2
Promoter
RBS
Coding Sequence
Terminator
Fault Tree and Hazard Table
Attack-Defense Tree and Attack-Defense Table
Identify Operational Environments
Phase 3
Semi-automated Assurance Case
Phase 4
Refactoring Assurance Case

Figure 7.3: Four phases of the TreAssure workflow integrating the fault tree and attack-defense tree with biological safety requirements to show mitigations are implemented in the assurance case. Mitigations are implemented in the design and code. An initial, semi-automated assurance case checks that there is sufficient evidence that the product satisfies the safety and security requirements for the identified operational environments. The semi-automated assurance case will be iteratively refactored because the application and environment typically evolve over time.

Two intermediate artifact tables are required to add rigor to the process of tracing hazards to mitigations and implementations. A generic overview of the tables is shown in Table 7.1, and the four phases of the TreAssure workflow is shown in Figure 7.3.

Table 7.1: Description of tables needed for TreAssure workflow.

Table Name	Lead Role	Column Headings	Purpose
Hazards Table	Safety Engineer	•Fault Nodes •Description •Mitigation Requirement •Responsible Agent(s) •Assurance Nodes	Identify key safety hazards to trace to mitigations and assurance nodes
Attack-Defense Table	Security Engineer	•Defense Nodes •Description •Responsible Agent(s) •Assurance Nodes	Identify key security defenses to trace to assurance nodes

7.3.1 Phase One

In phase one of Figure 7.3, three roles must be performed. The synthetic biologist identifies requirements for the SEBOs, the safety engineer identifies hazards, and the security engineer identifies attacks and mitigations. The information gathered in phase one is used in phase two.

7.3.2 Phase Two

Phase two also requires three roles. The synthetic biologist generates the design of the DNA parts appropriate for the desired SEBO behavior. The safety engineer generates a fault tree and hazards table described in Table 7.1. The hazards table identifies the fault nodes with their textual descriptions and assigns mitigation requirements along with a responsible agent. The trace from fault node to responsible agent can be a one-to-many relationship. The exact labels for the assurance nodes are not likely to be known, so that column could initially be blank or “TBD.” The security engineer produce an attack-defense table based upon the security analysis. The attack-defense table described in Table 7.1 identifies defense nodes with their textual descriptions and also assigns responsible agents. Again, exact labels for assurance nodes are not likely to be known at this time.

7.3.3 Phase Three

Phase three is a collaboration with all three engineers and other stakeholders such as project managers to evaluate an intermediate assurance case which could be developed with some automation. Every node identified in the first two tables should be initially populated in the assurance case as an evidence node. The nodes related to the fault tree should support a sub-goal of safety, and the nodes related to the attack-defense tree should support a sub-goal of security. The language of the attack-defense tree nodes can be copied into evidence nodes directly, but the language from the fault tree nodes will need to be negated. For example, “Safety system fails” in a fault node would become “Safety system does not fail” in an evidence node.

7.3.4 Phase Four

Phase four is an iterative process for refining or refactoring the assurance case to determine whether some evidence nodes can be combined, expanded, or marked as undeveloped. The text of the nodes can also be refined based upon updated experimental data or modified requirements.

At minimum, all fault nodes and mitigation nodes should lead to well developed evidence nodes within the assurance case. However, it could be appropriate for fault nodes to link to assumptions, justifications, or contexts, depending upon the operational environments. The tables can act as a checklist for confirming mitigations have been properly considered and implemented with trace links to the assurance case.

Any modification to one of the artifacts can then trigger warning flags for all of the linked nodes in the other artifacts for analysts to verify whether they have maintained correctness. For example, if the host organisms evolve or mutate to overcome a safety feature, it could mean that a mitigation is defeated or that a new fault is possible. These links facilitate cooperation across domains of expertise, and provide greater modularity and reuse for engineers. Importantly, they can provide regulators with improved coherence among the different artifacts needed to certify a cross-disciplinary, safety-critical system.

In the following section, we present a small case study to show how TreAssure works in practice.

7.4 TreAssure: Case Study

To evaluate the feasibility of TreAssure, consider the motivating example of engineering *E. coli* to reduce methane emissions combined with a temperature-sensitive kill switch (see Figure 7.2). To simplify the example, discussion is limited to safety and security of the kill-switch feature, recognizing that a full analysis of the system would be much more complicated. For example, it should also include analysis of the methane reduction system as well as supply-chain attacks related to DNA synthesis and food delivery.

The bacteria from Figure 7.2 are engineered to produce roughly the same amount of toxins and anti-toxins when at warmer temperatures (22° C-37° C), but an excess of toxins below 22° C [196]. This is an appropriate range for biological applications within mammalian digestive systems, to trigger termination after the cells are excreted. We simulated roles as the various engineers for this case study.

7.4.1 Phase One

The SEBO requirements are based on Figure 7.2, producing two main safety hazards: 1) the kill switch does not trigger when it should; and 2) the kill switch triggers when it should not. We similarly identified two main attacks: 1) causing

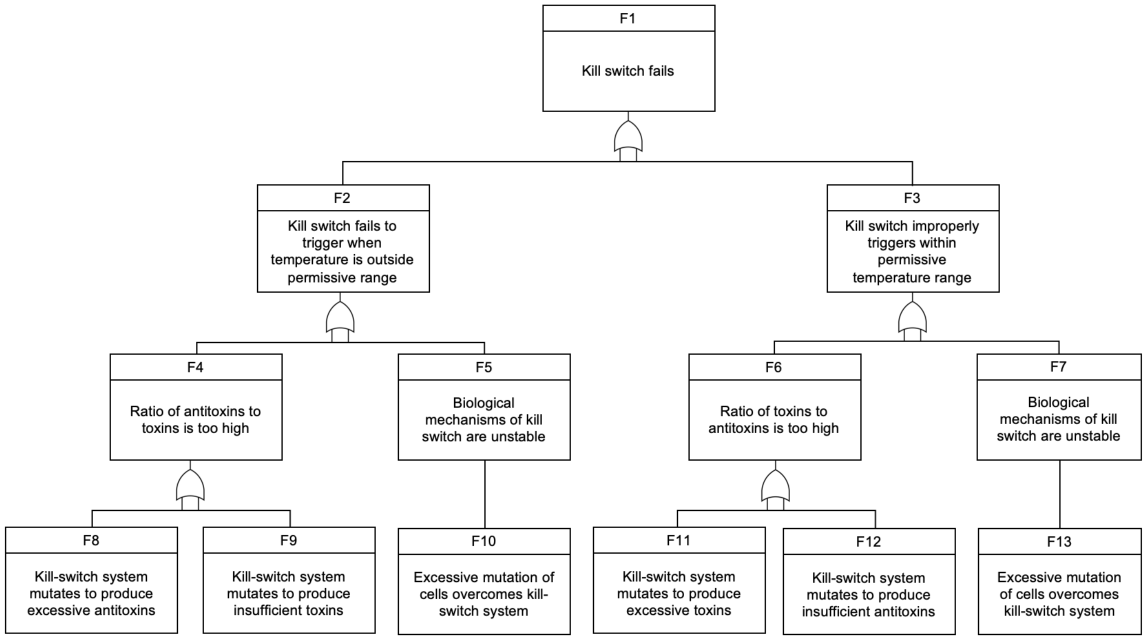

F1
Kill switch fails
F2
Kill switch fails to trigger when temperature is outside permissive range
F3
Kill switch improperly triggers within permissive temperature range
F4
Ratio of antitoxins to toxins is too high
F5
Biological mechanisms of kill switch are unstable
F6
Ratio of toxins to antitoxins is too high
F7
Biological mechanisms of kill switch are unstable
F8
Kill-switch system mutates to produce excessive antitoxins
F9
Kill-switch system mutates to produce insufficient toxins
F10
Excessive mutation of cells overcomes kill-switch system
F11
Kill-switch system mutates to produce excessive toxins
F12
Kill-switch system mutates to produce insufficient antitoxins
F13
Excessive mutation of cells overcomes kill-switch system

Figure 7.4: Fault tree for the cryodeath kill switch.

the kill switch to trigger when it should not; and 2) preventing the kill switch from triggering when it should.

7.4.2 Phase Two

We produced the fault tree in Figure 7.4 and the attack-defense tree in Figure 7.5. The two main hazards are represented as nodes F2 and F3 in Figure 7.4. The specific causes to be mitigated are the leaf nodes of F8-F13 populated in Table 7.2, indicating that either the DNA of the kill switch itself can mutate within the cells, or the cells themselves can mutate to overcome the kill switch as a population by not passing its code to its children. This a natural biological process because cells do not normally contain kill switches, and to maintain them cells must expend resources which otherwise could be dedicated to survival and reproduction.

We then populate the attack-defense table shown in Table 7.3 based on the attacks and mitigations from attack-defense tree from Figure 7.5. Populating the

Table 7.2: Hazards traced to mitigation requirements.

Fault Nodes	Description	Mitigation Requirement	Responsible Agent(s) for Planned Mitigation	Assurance Nodes
F8	Kill-switch system mutates to produce excessive antitoxins	Utilize stable organisms	DNA synthesis facility; Modeling software; Lab technicians; Regulatory Agencies	G3, E9
F9	Kill-switch system mutates to produce insufficient toxins	Utilize stable organisms	DNA synthesis facility; Modeling software; Lab technicians; Regulatory Agencies	G3, E10
F10 & F13	Excessive mutation of cells overcomes kill-switch system	Utilize stable organisms	DNA synthesis facility; Modeling software; Lab technicians; Regulatory Agencies	G3, E11, E14
F11	Kill-switch system mutates to produce excessive toxins	Utilize stable organisms	DNA synthesis facility; Modeling software; Lab technicians; Regulatory Agencies	G3, E12
F12	Kill-switch system mutates to produce insufficient antitoxins	Utilize stable organisms	DNA synthesis facility; Modeling software; Lab technicians; Regulatory Agencies	G3, E13

table could involve discussions with a synthetic biologist to determine whether the mitigation requirements are reasonable or feasible. The agents identified as responsible for the planned mitigation can be people, software or hardware processes, or regulatory agencies which ultimately provide evidence to support requirements have been met. Repetition of a mitigation requirement or responsible agent suggests later groupings and refinements for the assurance case. The last column for the Assurance Nodes will initially be blank or “TBD.”

Our scenario assumes there are two distinct facilities involved, each with their own vulnerabilities. One is the feeding facility where the cattle are handled, and the other is the DNA synthesis facility where the organisms are designed and engineered for delivery to the feeding facility. An attacker would need either physical or logical access to one of the facilities, and could either be an internal employee or external actor. Nodes M1-M8 describe various defense measures related to physical and logical security, but also include more nuanced considerations such as employee and client background checks.

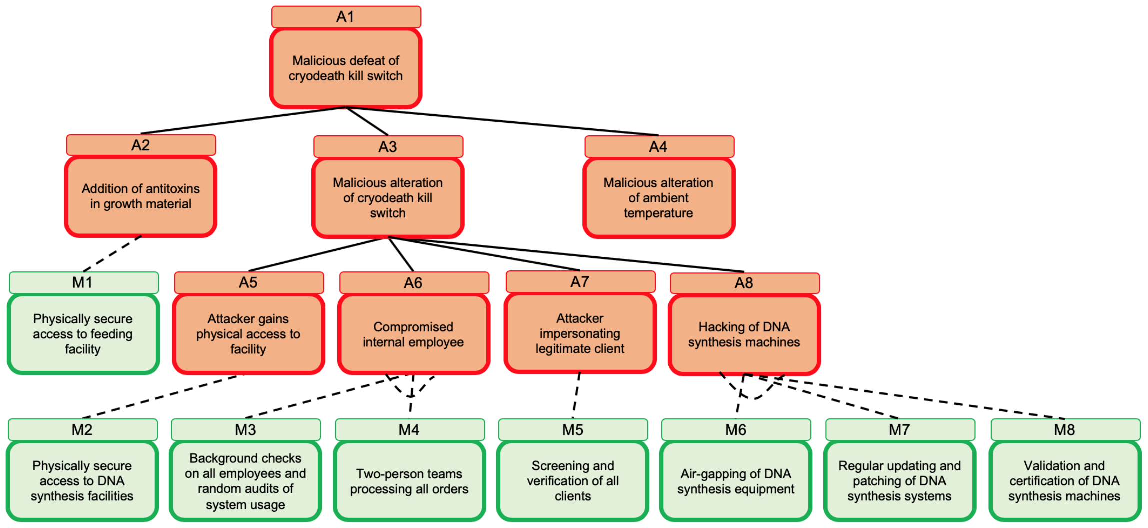

A1
Malicious defeat of cryodeath kill switch
A2
Addition of antitoxins in growth material
A3
Malicious alteration of cryodeath kill switch
A4
Malicious alteration of ambient temperature
M1
Physically secure access to feeding facility
A5
Attacker gains physical access to facility
A6
Compromised internal employee
A7
Attacker impersonating legitimate client
A8
Hacking of DNA synthesis machines
M2
Physically secure access to DNA synthesis facilities
M3
Background checks on all employees and random audits of system usage
M4
Two-person teams processing all orders
M5
Screening and verification of all clients
M6
Air-gapping of DNA synthesis equipment
M7
Regular updating and patching of DNA synthesis systems
M8
Validation and certification of DNA synthesis machines

Figure 7.5: Attack-defense tree for the cryodeath kill switch.

Table 7.3: Defenses traced to mitigation requirements.

Defense Nodes	Description	Responsible Agent(s) for Planned Mitigation	Assurance Nodes
M1	Physically secure access to feeding facility	Feeding facility maintenance manager	G2, E1
M2	Physically secure access to DNA synthesis facilities	DNA facility maintenance manager	G2, E2
M3	Background checks on all employees and random audits of system usage	DNA facility's HR and IT managers	G2, E3
M4	Two-person teams processing all orders	DNA facility's project manager	G2, E4
M5	Screening and verification of all clients	DNA facility's project manager	G2, E5
M6	Air-gapping of DNA synthesis equipment	DNA facility's CIO	G2, E6
M7	Regular updating and patching of DNA synthesis systems	DNA synthesis facility's CIO	G2, E7
M8	Validation and certification of DNA synthesis machines	DNA synthesis facility's CIO	G2, E8

7.4.3 Phase Three

Next, we assess the intermediate assurance case, which can have semi-automated sub-goals for safety and security, and one evidence node for every leaf node from the fault tree and attack-defense tree. One possible intermediate assurance case is shown in Figure 7.6. The green nodes on the left side are automatically developed from M1-M8 in the attack-defense tree, and the red nodes on the right side represent negations of the fault-tree nodes of F8-F13.

7.4.4 Phase Four

We manually collapsed some evidence nodes, with guidance provided by the trace tables where repetition of mitigation requirements or responsible agents suggest which nodes are good candidates for combination. For example, E6 and E7 might combine into E7-1. It is also possible that some of the nodes should be expanded into multiple evidence nodes, such as E3 expanding into E3-1 and E4-1. Some nodes might have a many-to-one relationship such as nodes E9-E14 becoming nodes E9-1 and E10-1. It is also possible for some nodes to be deleted or become undeveloped. The assurance case was built using the software tool AdvoCATE from Denney and Pai [50].

Figure 7.7 is our finalized assurance case after iterating over evidence nodes and identifying which can be combined. We then can finalize the assurance node columns from Tables 7.2 and 7.3.

A regulator should thus be able to review all three of the artifacts, along with an associated tables, to readily see that all of the faults have been properly and adequately mitigated. The fault-tree analysis identifies actionable hazards and the safety requirements to prevent them. The attack-defense tree analysis identifies actionable vulnerabilities and the requirements needed to prevent them. The

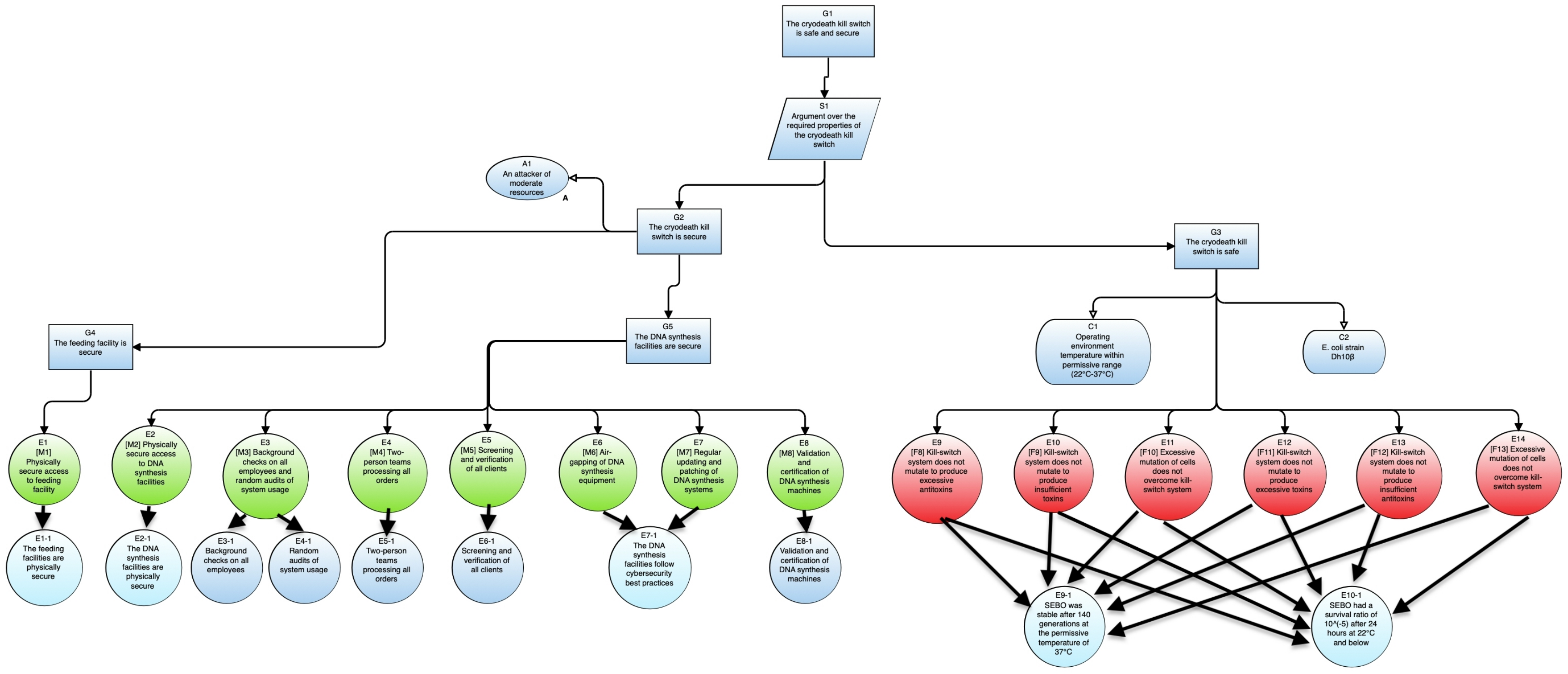

G1 The cryodeath kill switch is safe and secure
S1 Argument over the required properties of the cryodeath kill switch
A1 An attacker of moderate resources
A
G2 The cryodeath kill switch is secure
G3 The cryodeath kill switch is safe
G4 The feeding facility is secure
G5 The DNA synthesis facilities are secure
C1 Operating environment temperature within permissive range (22°C-37°C)
C2 E. coli strain Dh10β
E1 [M1] Physically secure access to feeding facility
E2 [M2] Physically secure access to DNA synthesis facilities
E3 [M3] Background checks on all employees and random audits of system usage
E4 [M4] Two-person teams processing all orders
E5 [M5] Screening and verification of all clients
E6 [M6] Air-gapping of DNA synthesis equipment
E7 [M7] Regular updating and patching of DNA synthesis systems
E8 [M8] Validation and certification of DNA synthesis machines
E9 [F8] Kill-switch system does not mutate to produce excessive antitoxins
E10 [F9] Kill-switch system does not mutate to produce insufficient toxins
E11 [F10] Excessive mutation of cells does not overcome kill-switch system
E12 [F11] Kill-switch system does not mutate to produce excessive toxins
E13 [F12] Kill-switch system does not mutate to produce insufficient antitoxins
E14 [F13] Excessive mutation of cells does not overcome kill-switch system
E1-1 The feeding facilities are physically secure
E2-1 The DNA synthesis facilities are physically secure
E3-1 Background checks on all employees
E4-1 Random audits of system usage
E5-1 Two-person teams processing all orders
E6-1 Screening and verification of all clients
E7-1 The DNA synthesis facilities follow cybersecurity best practices
E8-1 Validation and certification of DNA synthesis machines
E9-1 SEBO was stable after 140 generations at the permissive temperature of 37°C
E10-1 SEBO had a survival ratio of 10^(-5) after 24 hours at 22°C and below

Figure 7.6: A possible semi-automated assurance case with every leaf node from the fault tree and attack-defense tree corresponding to an evidence node. The bottom row represents one possible iteration of refactoring the evidence nodes.

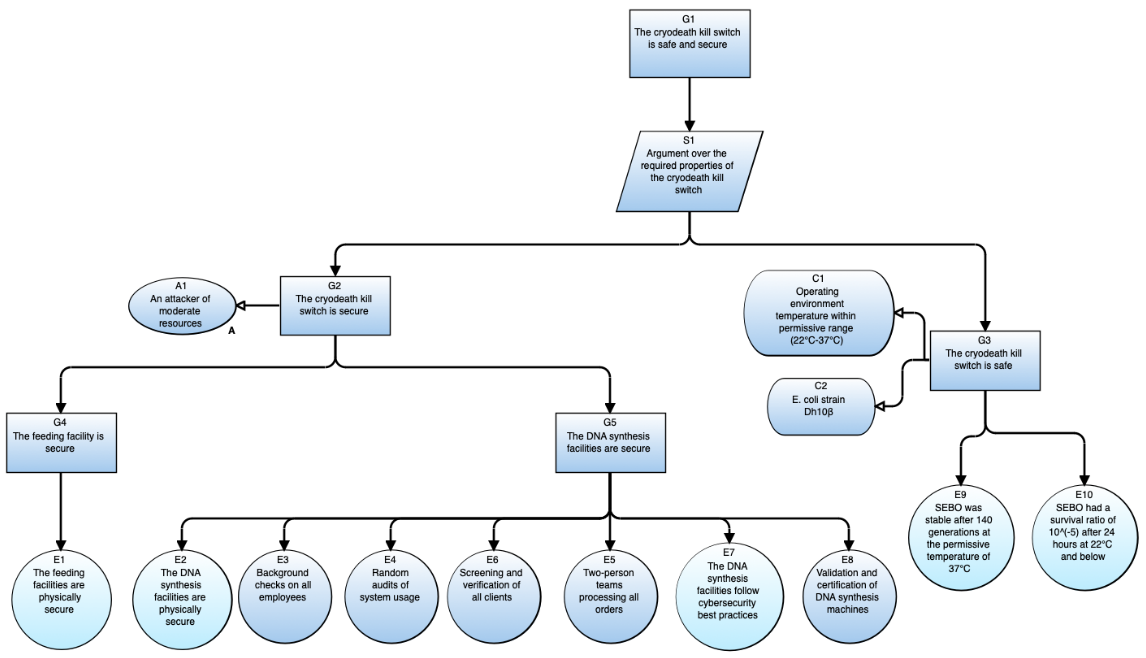

G1
The cryodeath kill switch is safe and secure
S1
Argument over the required properties of the cryodeath kill switch
A1
An attacker of moderate resources
A
G2
The cryodeath kill switch is secure
C1
Operating environment temperature within permissive range (22°C-37°C)
G3
The cryodeath kill switch is safe
C2
E. coli strain Dh10β
G4
The feeding facility is secure
G5
The DNA synthesis facilities are secure
E9
SEBO was stable after 140 generations at the permissive temperature of 37°C
E10
SEBO had a survival ratio of 10^(-5) after 24 hours at 22°C and below
E1
The feeding facilities are physically secure
E2
The DNA synthesis facilities are physically secure
E3
Background checks on all employees
E4
Random audits of system usage
E6
Screening and verification of all clients
E5
Two-person teams processing all orders
E7
The DNA synthesis facilities follow cybersecurity best practices
E8
Validation and certification of DNA synthesis machines

Figure 7.7: A possible final assurance case after iteration.

assurance case argues there is sufficient evidence to trust that the system is adequately protected against these hazards and threats for the specified environment. In other words, the first two artifacts help identify what needs to be done, and the assurance case shows how the requirements have been met.

7.5 Limitations

A key limitation of this study is that we simulated roles as the engineers, and the TreAssure workflow should be tested in a more realistic setting to determine its usefulness and effectiveness. Also, it was only applied to this kill-switch system, but it is nevertheless representative of other SB systems. We leave a more complete evaluation as future work.

7.6 Related Work

Johnson and Kelly originally proposed a method for linking fault trees to attack-defense trees [111], and there has been some prior work on applying assurance cases to SB. Cohen, et al., proposed the use of a type of dynamic assurance case, the assurance timeline to incorporate evolution into the assurance case [44]. In follow-on work, Firestone and Cohen suggested using assurance recipes to help end-users build assurance cases [70]. This chapter differs in that it proposes combining fault trees and attack trees to build a traceable assurance case.

There has been recent research on the importance of facilitating assurance case development through well-formed patterns [53, 70]. There are now a number of tools devoted to assessing assurance cases [146], including analysis of safety architectures in bowtie diagrams which build links to hazard tables [50, 52].

There is a robust history of threat modeling, dating as far back as 1977 with the concept of software architectural patterns introduced by Christopher Alexander [193]. The core concepts were not introduced until 1994 by Edward Amaroso, who introduced threat trees based on decision tree diagrams [5]. In 1998, a research group with famed security expert Bruce Schneier introduced the concept of attack trees in a seminal paper titled “Toward a Secure System Engineering Methodology.” [187]

There are now several mature methodologies for threat modeling, each with their respective benefits and drawbacks. The most commonly applied framework, known by the acronym STRIDE, was developed at Microsoft [193]. STRIDE stands for: Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Escalation of privilege. The opposite qualities are those which should be designed into the system: authenticity, integrity, non-repudiation, confidentiality, availability, and authorization [193].

Threat modeling in specialized domains can present specialized challenges. Medical devices present special challenges because they require regulatory approval, but “security is a relatively new concern for regulatory bodies; bug-averse manufacturers have traditionally had little incentive to add security mechanisms that might cause problems or slow down regulatory approval.” [30]

7.7 Conclusion and Future Work

This chapter has presented TreAssure, a semi-automated method for leveraging fault trees and attack-defense trees to help develop assurance cases. All three artifacts have their own benefits, but together they can help stakeholders from research, industry, and government better certify claims regarding the safety and security of novel SB applications.

One area for future work involves gathering qualitative and quantitative feedback from stakeholders using this new workflow in practice for a real-world SEBO project. Another area involves tool development to automate generation of the assurance case from the fault trees, attack-defense trees, and the associated tables.

Chapter 8

Conclusion and Future Work

This dissertation has made the argument that SB is a technology with amazing potential. It could lead to renewable energy, cures for diseases, and even outer-space exploration. Such a powerful technology, of course, could have disastrous consequences through negligence or malicious intent. Engineered organisms can evolve and mutate, making it especially difficult to validate and verify their behavior.

As an engineering discipline, SB is centered upon programming life through rational, methodical manipulation of DNA sequences, representing a vast configuration space. Traditional software engineering principles could be applied to SB to help all stakeholders reason about novel designs, and interdisciplinary problems will have interdisciplinary solutions.

The goal of this dissertation is not to conjure up worst-case scenarios, but to encourage responsible, effective, and beneficial innovation. There is a distinct lack of meaningful regulation of SB at both the national and international level, and any new regulatory frameworks must adequately protect the public and the environment without stifling legitimate research.

Chapter 3 demonstrates how the software engineering concept of assurance cases could help synthetic biologists consider safety earlier in their design-build-test cycle. Assurance cases are commonly used in safety-critical systems such as avionics, offshore drilling, and nuclear power, but the concept could be extended to emerging domains such as SB.

Chapter 4 shows how the software engineering concept of feature models and software product lines could help synthetic biologists reason about the vast configuration space available for programming DNA sequences. As a case study, domain knowledge about safety features is used to populate a partial feature model to help visualize which components are required to build valid and complete kill switches with specific, desired characteristics.

Chapter 5 is an examination of what makes SB difficult to regulate, but nevertheless argues there needs to be a renewed international effort to develop meaningful oversight to ensure safety and security for SEBOs. Traditional hard laws such as treaties, statutes, and regulations are likely to be inadequate or stifle legitimate innovation. Soft-law solutions are more likely to effectively establish meaningful protections for the public and the environment, and all stakeholders should be a part of the process, including members from industry, academia, and DIY groups. Software engineers and cybersecurity experts should specifically be targeted for inclusion, because they can better understand the threats associated with the software and hardware used in the SB design-build-test cycle.

Chapter 6 describes a lightweight tool to analyze DNA sequences at both the nucleotide and peptide level, specifically toxin and non-toxin sequences. The output can then be visualized in a multitude of ways, and also used as input for pre-existing toxin classification tools.

Lastly, Chapter 7 shows a proposed certification process for regulating SB by integrating a fault tree and attack-defense tree into an assurance case as part of an iterative process to help identify unacceptable threats or safety risks. The process provides traceabaility across all three artifacts to improve overall safety and security during the design-build-test cycle. The final products are inherently interdisciplinary and collaboration among various experts will improve readability, coherence, and consistency for regulators.

8.1 Future Work

There is ample opportunity to explore future work based on the ideas presented in this dissertation. The assurance recipes from Chapter 3 could be developed into an interactive software system to help iGEM teams build assurance cases. A preliminary interactive safety-feature system was developed by the 2018 University of Nebraska–Lincoln iGEM team to fulfill collaboration requirements [250].

The feature models from Chapter 4 could lead to a new domain specific-language for generating SPL constraints. This would need to be evaluated in practice with teams of synthetic biologists, perhaps also for the iGEM Competition. Using feature models for SB systems presents challenges, some of which are typical for SPLs such as scalability, and collaborative workflows. They also present novel challenges, such as how to represent dynamic elements related to biological evolution and mutation.

The AlphaFreque tool from Chapter 6, by itself, only provides more data for analysis. The data from all the different counts, both for peptides and nucleotides of different lengths, could be fed into existing prediction tools to determine whether it improves protein prediction accuracy by increasing the length of the k-mers analyzed. There are ever-increasing methods for analyzing large amount of data, and it remains to determine which methods best characterize toxins versus non-toxins.

Notably, prior work has explicitly assumed that variable-length subsequences are important for determining protein characteristics: “it is unrealistic to assume that fixed-length k-mers are units of biological sequences and that more meaningful units need to be introduced.” [8] However, there is some evidence that variable-length analyses do not significantly improve protein classification, suggesting that “k-mer representation [fixed-length motif] is a tough-to-beat baseline in protein classification problems.” [8] Another method would be to train an image classifier based on subsets of the toxins and non-toxins. This would likely be easiest using the 2-D format shown in Figures 6.19 and 6.21.

There is ample opportunity for developing automation of some of the TreAssure process from Chapter 7 by extending existing software tools. With the suggested tables populated with hazards and mitigations, an Eclipse plugin or extension of AdvoCATE [53] could generate template stubs for an assurance case, possibly using “assurance recipes” based on synthetic biology domain knowledge [70]. The traceability links could be automatically generated such that modifications would flag all connected nodes for re-validation. Ideally, all three tables and all three artifacts would be contained within one software tool for improved compatibility and file sharing. Another improvement would be to add predicate logic, probabilities, and severities to TreAssure for further automated analysis.

Lastly, implementation of the TreAssure workflow from Chapter 7 as a soft-law best practice for research, industry, and government could help alleviate some of the regulatory concerns raised in Chapter 5.

References

[1] Adam, L., Kozar, M., Letort, G., Mirat, O., Srivastava, A., Stewart, T., Wilson, M. L., and Peccoud, J. Strengths and limitations of the federal guidance on synthetic DNA. *Nature Biotechnology 29*, 3 (2011), 208.

[2] Ahteensuu, M. Synthetic biology, genome editing, and the risk of bioterrorism. *Science and engineering ethics 23*, 6 (2017), 1541–1561.

[3] Akyildiz, I., Pierobon, M., Balasubramaniam, S., and Koucheryavy, Y. The internet of bio-nano things. *Communications Magazine, IEEE 53*, 3 (March 2015), 32–40.

[4] Aldhous, P. The bioweapon is in the post. *New Scientist 188*, 2525 (2005), 8–9.

[5] Amoroso, E. G. *Fundamentals of computer security technology*. Prentice-Hall, Inc., 1994.

[6] Andres, J., Blomeier, T., and Zurbriggen, M. D. Synthetic switches and regulatory circuits in plants. *Plant Physiology 179*, 3 (2019), 862–884.

[7] Apweiler, R., Bairoch, A., Wu, C. H., Barker, W. C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., et al. UniProt: the universal protein knowledgebase. *Nucleic acids research 32*, suppl_1 (2004), D115–D119.

[8] ASGARI, E., MCHARDY, A. C., AND MOFRAD, M. R. Probabilistic variable-length segmentation of protein sequences for discriminative motif discovery (DiMotif) and sequence embedding (ProtVecX). *Scientific Reports 9*, 1 (2019), 3577.

[9] ASGARI, E., AND MOFRAD, M. R. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one 10*, 11 (2015), e0141287.

[10] ASGARI, E., MÜNCH, P. C., LESKER, T. R., MCHARDY, A. C., AND MOFRAD, M. R. Nucleotide-pair encoding of 16s rRNA sequences for host phenotype and biomarker detection. *bioRxiv* (2018), 334722.

[11] BALDWIN, G. *Synthetic biology: A primer*. World Scientific, 2016.

[12] BALDWIN, G., DICKINSON, R., AND KITNEY, R. I. *Synthetic biology-a primer: revised edition*. Imperial College Press, 2015.

[13] BANG, C., FIEDLER, J., AND THUM, T. Cardiovascular importance of the microRNA-23/27/24 family. *Microcirculation 19*, 3 (2012), 208–214.

[14] BAUMGAERTNER, E. It was sensitive data from a U.S. anti-terror program and terrorists could have gotten to it for years, records show. `latimes.com/science/sciencenow/la-sci-biowatch-20190402-story.html`, 2019.

[15] BEGLEY, S. Fertility clinics around the world asked 'CRISPR babies' scientist for how-to help. `statnews.com/2019/05/28/fertility-clinics-asked-crispr-babies-scientist-for-how-to-help/`, 2019.

[16] BEGLEY, S. Scientists call for do-over for rules on creating 'CRISPR babies'. `scientificamerican.com/article/`

`scientists-call-for-do-over-for-rules-on-creating-crispr-babies/`, 2019.

[17] Benavides, D., Segura, S., and Ruiz-Cortés, A. Automated analysis of feature models 20 years later: A literature review. *Information systems 35*, 6 (2010), 615–636.

[18] Benavides, D., Segura, S., and Ruiz-Cortés, A. Automated analysis of feature models 20 years later: A literature review. *Inf. Syst. 35*, 6 (Sept. 2010), 615–636.

[19] Bereza-Malcolm, L. T., Mann, G., and Franks, A. E. Environmental sensing of heavy metals through whole cell microbial biosensors: a synthetic biology approach. *ACS synthetic biology 4*, 5 (2014), 535–546.

[20] Berube, D. M. *Mosquitoes Bite: A Zika Story of Vector Management and Gene Drives*. Springer International Publishing, 2020, pp. 143–163.

[21] Beuche, D., Papajewski, H., and Schröder-Preikschat, W. Variability management with feature models. *Science of Computer Programming 53*, 3 (2004), 333–352.

[22] Bloomfield, R., and Netkachova, K. Building blocks for assurance cases. In *Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on* (Nov 2014), pp. 186–191.

[23] Bornholt, J., Lopez, R., Carmean, D. M., Ceze, L., Seelig, G., and Strauss, K. A DNA-based archival storage system. *ACM SIGOPS Operating Systems Review 50*, 2 (2016), 637–649.

[24] Bostrom, N. Existential risks: Analyzing human extinction scenarios and related hazards. *Journal of Evolution and Technology 9* (2002).

[25] BOSTROM, N. The vulnerable world hypothesis. *Global Policy* (2019).

[26] BRODWIN, E. Silicon valley startups backed by celebrities like bill gates are using gene-editing tool CRISPR to make meat without farms and to disrupt a $200 billion industry. `www.businessinsider.com/silicon-valley-startups-using-crispr-chicken-beef-memphis-meats/new-age-meats-2019-3`, 2019.

[27] BROWN, K. V. It wasn't biohacking that killed the biohacker: He drowned. `bloomberg.com/news/articles/2018-06-29/autopsy-reveals-biohacker-traywick-died-from-accidental-drowning`, 2018.

[28] BROWN, K. V. What does an infamous biohacker's death mean for the future of DIY science? `theatlantic.com/science/archive/2018/05/aaron-traywick-death-ascendance-biomedical/559745/`, 2018.

[29] BURGESS-BROWN, N. A., SHARMA, S., SOBOTT, F., LOENARZ, C., OPPERMANN, U., AND GILEADI, O. Codon optimization can improve expression of human genes in Escherichia coli: a multi-gene study. *Protein expression and purification 59*, 1 (2008), 94–102.

[30] BURLESON, W., CLARK, S. S., RANSFORD, B., AND FU, K. Design challenges for secure implantable medical devices. In *Proceedings of the 49th Annual Design Automation Conference* (2012), pp. 12–17.

[31] CAI, Y., WILSON, M. L., AND PECCOUD, J. GenoCAD for iGEM: a grammatical approach to the design of standard-compliant constructs. *Nucleic acids research 38*, 8 (2010), 2637–2644.

[32] CALIFORNIA STATE LEGISLATURE. Sb180. `leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201920200SB180`, 2019.

[33] CAMERON, D. E., BASHOR, C. J., AND COLLINS, J. J. A brief history of synthetic biology. *Nature Reviews Microbiology 12*, 5 (2014), 381.

[34] CARDOSO, J. G., JENSEN, K., LIEVEN, C., LÆRKE HANSEN, A. S., GALKINA, S., BEBER, M., OZDEMIR, E., HERRGÅRD, M. J., REDESTIG, H., AND SONNENSCHEIN, N. Cameo: a python library for computer aided metabolic engineering and optimization of cell factories. *ACS synthetic biology 7*, 4 (2018), 1163–1166.

[35] CASHMAN, M., FIRESTONE, J., COHEN, M. B., THIANNIWET, T., AND NIU, W. Dna as features: Organic software product lines. In *Proceedings of the 23rd International Systems and Software Product Line Conference - Volume A* (New York, NY, USA, 2019), SPLC '19, ACM, pp. 108–118.

[36] CDC AND USDA. Federal select agents program. `selectagents.gov`, 2017.

[37] CHAKRABORTY, A., GHOSH, S., CHOWDHARY, G., MAULIK, U., AND CHAKRABARTI, S. DBETH: a database of bacterial exotoxins for human. *Nucleic Acids Research 40*, D1 (2011), D615–D620.

[38] CIMPANU, C. AT&T employees took bribes to plant malware on the company's network. `zdnet.com/article/at-t-employees-took-bribes-to-plant-malware-on-the-companys-network`, 2019.

[39] CIMPANU, C. Mysterious iranian group is hacking into DNA sequencers. `www.zdnet.com/article/mysterious-iranian-group-is-hacking-into-dna-sequencers/`, 2019.

[40] CLARKE, R., AND VALENTIN, J. The history of ICRP and the evolution of its policies. *Annals of the ICRP 39*, 1 (2009), 75–110.

[41] Clegg, K., Li, M., Stamp, D., Grigg, A., and McDermid, J. A sysml profile for fault trees - linking safety models to system design. In *Computer Safety, Reliability, and Security - 38th International Conference, SAFECOMP 2019, Turku, Finland, September 11-13, 2019, Proceedings* (2019), pp. 85–93.

[42] Cleland-Huang, J., Vierhauser, M., and Bayley, S. Dronology: An incubator for cyber-physical systems research. In *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results* (2018), ICSE-NIER '18, pp. 109–112.

[43] Clements, P., and Northrop, L. *Software Product Lines: Practices and Patterns*. Addison-Wesley, Boston, 2002.

[44] Cohen, M. B., Firestone, J., and Pierobon, M. The assurance timeline: Building assurance cases for synthetic biology. In *Computer Safety, Reliability, and Security* (Cham, 2016), A. Skavhaug, J. Guiochet, E. Schoitsch, and F. Bitsch, Eds., Springer International Publishing, pp. 75–86.

[45] Conmy, P., and Bate, I. Assuring safety for component based software engineering. In *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering* (Jan 2014), pp. 121–128.

[46] Crick, F., et al. Central dogma of molecular biology. *Nature 227*, 5258 (1970), 561–563.

[47] Daniel, R., Rubens, J. R., Sarpeshkar, R., and Lu, T. K. Synthetic analog computation in living cells. *Nature 497*, 7451 (2013), 619.

[48] DAVIS, N. First human-monkey chimera raises concern among scientists. `theguardian.com/science/2019/aug/03/first-human-monkey-chimera-raises-concern-among-scientists`, 2019.

[49] DEL VECCHIO, D., QIAN, Y., MURRAY, R. M., AND SONTAG, E. D. Future systems and control research in synthetic biology. *Annual Reviews in Control* (2018).

[50] DENNEY, E., AND PAI, G. Tool support for assurance case development. *Autom. Softw. Eng. 25*, 3 (2018), 435–499.

[51] DENNEY, E., PAI, G., AND WHITESIDE, I. Formal foundations for hierarchical safety cases. 52–59.

[52] DENNEY, E., PAI, G., AND WHITESIDE, I. The role of safety architectures in aviation safety cases. *Reliability Engineering & System Safety 191* (2019), 106502.

[53] DENNEY, E. W., AND PAI, G. J. Safety case patterns: theory and applications. *NASA* (2015).

[54] DIY BIO. DIY Bio draft codes of ethics. `diybio.org/codes/`, 2011.

[55] DOXEY, A. C., MANSFIELD, M. J., AND MONTECUCCO, C. Discovery of novel bacterial toxins by genomics and computational biology. *Toxicon 147* (2018), 2 – 12. Basic science and clinical aspects of botulinum and other toxins.

[56] DUNLAP, G., AND PAUWELS, E. The intelligent and connected bio-labs of the future. *The Wilson Center* (2017).

[57] EBERBACH, E., GOLDIN, D., AND WEGNER, P. *Turing's ideas and models of computation*. Springer, 2004.

[58] EDGE, K., RAINES, R., GRIMAILA, M., BALDWIN, R., BENNINGTON, R., AND REUTER, C. The use of attack and protection trees to analyze security for an online banking system. In *2007 40th Annual Hawaii International Conference on System Sciences* (Jan 2007), pp. 144b–144b.

[59] EHRENBERG, R. Engineered yeast paves way for home-brew heroin. *Nature News 521*, 7552 (2015), 267.

[60] ELLIS, S. J., KLINGE, T. H., LATHROP, J. I., LUTZ, J. H., LUTZ, R. R., MINER, A. S., AND POTTER, H. D. Runtime fault detection in programmed molecular systems. *ACM Trans. Softw. Eng. Methodol. 28*, 2 (2019), 6:1–6:20.

[61] ELOWITZ, M. B., AND LEIBLER, S. A synthetic oscillatory network of transcriptional regulators. *Nature 403* (2000), 335–338.

[62] ENDY, D. Foundations for engineering biology. *Nature 438*, 7067 (2005), 449.

[63] EVANS, N. G., AND SELGELID, M. J. Biosecurity and open-source biology: The promise and peril of distributed synthetic biological technologies. *Science and engineering ethics* (2014), 1–19.

[64] FAGES, F., LE GULUDEC, G., BOURNEZ, O., AND POULY, A. Strong turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs. In *Computational Methods in Systems Biology* (Cham, 2017), J. Feret and H. Koeppl, Eds., Springer International Publishing, pp. 108–127.

[65] FDA. Information about self-administration of gene therapy. `fda.gov/BiologicsBloodVaccines/CellularGeneTherapyProducts/ucm586343.htm`, 2017.

[66] FDA. FDA approves the first drug with an indication for treatment of smallpox. `fda.gov/NewsEvents/Newsroom/PressAnnouncements/ucm613496.htm`, 2018.

[67] FEINBERG, M. Chemical reaction network structure and the stability of complex isothermal reactors i. the deficiency zero and deficiency one theorems. *Chemical Engineering Science 42* (01 1987), 2229–2268.

[68] FENG, L., KING, A. L., CHEN, S., AYOUB, A., PARK, J., BEZZO, N., SOKOLSKY, O., AND LEE, I. A safety argument strategy for PCA closed-loop systems: A preliminary proposal. In *MCPS'14* (2014), pp. 94–99.

[69] FIRESTONE, J. The need for soft law to regulate synthetic biology. *Jurimetrics 60* (2020), 139–173.

[70] FIRESTONE, J., AND COHEN, M. B. The assurance recipe: Facilitating assurance patterns. In *Computer Safety, Reliability, and Security* (Cham, 2018), B. Gallina, A. Skavhaug, E. Schoitsch, and F. Bitsch, Eds., Springer International Publishing, pp. 22–30.

[71] FOKKER, J. Organizations leave backdoors open to cheap remote desktop protocol attacks. `securingtomorrow.mcafee.com/mcafee-labs/organizations-leave-backdoors-open-to-cheap-remote-desktop-protocol-attacks/` 2018.

[72] FOR BIOSECURITY, C., AND BIOPREPAREDNESS. An efficient and practical approach to biosecurity. `biosecurity.dk/biosecuritybook/`, 2019.

[73] FORUM, W. E. Biosecurity innovation and risk reduction: A global framework for accessible, safe and secure dna synthesis.

[74] FRAZAR, S. L., HUND, G. E., BONHEYO, G. T., DIGGANS, J., BARTHOLOMEW, R. A., GEHRIG, L., AND GREAVES, M. Defining the synthetic biology supply chain. *Health security 15*, 4 (2017), 392–400.

[75] FROW, E. From experiments of concern to groups of concern constructing and containing citizens in synthetic biology. *Science, Technology, & Human Values* (2017), 1–27.

[76] FUSSELL, J. Fault tree analysis: concepts and techniques. In *Pressure vessels and piping: design and analysis. IV*. 1976.

[77] GACESA, R., BARLOW, D. J., AND LONG, P. F. Machine learning can differentiate venom toxins from other proteins having non-toxic physiological functions. *PeerJ Computer Science 2* (2016), e90.

[78] GAGE, P. A new algorithm for data compression. *The C Users Journal 12*, 2 (1994), 23–38.

[79] GALANIE, S., THODEY, K., TRENCHARD, I. J., INTERRANTE, M. F., AND SMOLKE, C. D. Complete biosynthesis of opioids in yeast. *Science 349*, 6252 (2015), 1095–1100.

[80] GALDZICKI, M., CLANCY, K. P., OBERORTNER, E., POCOCK, M., QUINN, J. Y., RODRIGUEZ, C. A., ROEHNER, N., WILSON, M. L., ADAM, L., ANDERSON, J. C., ET AL. The synthetic biology open language (SBOL) provides a community standard for communicating designs in synthetic biology. *Nature biotechnology 32*, 6 (2014), 545.

[81] GALLEGOS, J. E., BOYER, C., PAUWELS, E., KAPLAN, W. A., AND PECCOUD, J. The open insulin project: A case study for ‘biohacked’ medicines. *Trends in biotechnology* (2018).

[82] GAMAARACHCHI, H., PARAMESWARAN, S., AND SMITH, M. A. Featherweight long read alignment using partitioned reference indexes. *Scientific reports 9*, 1 (2019), 4318.

[83] GARDNER, T. S., CANTOR, C. R., AND COLLINS, J. J. Construction of a genetic toggle switch in Escherichia coli. *Nature 402* (2000), 339–342. One of two initial pubs on synthetic biology devices.

[84] GARFINKLE, M., AND KNOWLES, L. *Synthetic biology, biosecurity, and biosafety*. Springer, 2014.

[85] GARVIN, B. J., AND COHEN, M. B. Feature interaction faults revisited: An exploratory study. In *2011 IEEE 22nd International Symposium on Software Reliability Engineering* (2011), IEEE, pp. 90–99.

[86] GARVIN, B. J., COHEN, M. B., AND DWYER, M. B. Evaluating improvements to a meta-heuristic search for constrained interaction testing. *Empirical Software Engineering 16*, 1 (2011), 61–102.

[87] GIBSON, D. G., GLASS, J. I., LARTIGUE, C., NOSKOV, V. N., CHUANG, R.-Y., ALGIRE, M. A., BENDERS, G. A., MONTAGUE, M. G., MA, L., MOODIE, M. M., ET AL. Creation of a bacterial cell controlled by a chemically synthesized genome. *Science 329*, 5987 (2010), 52–56.

[88] GIBSON, D. G., YOUNG, L., CHUANG, R.-Y., VENTER, J. C., HUTCHISON III, C. A., AND SMITH, H. O. Enzymatic assembly of dna molecules up to several hundred kilobases. *Nature methods 6*, 5 (2009), 343.

[89] GREENBERG, A. A landmark legal shift opens pandora's box for DIY guns. `wired.com/story/a-landmark-legal-shift-opens-pandoras-box-for-diy-guns/`, 2018.

[90] GUAN, Z.-J., SCHMIDT, M., PEI, L., WEI, W., AND MA, K.-P. Biosafety considerations of synthetic biology in the international genetically engineered machine (iGEM) competition. *BioScience 63*, 1 (2013), 25–34.

[91] HART, G., AND RUDMAN, W. B. New world coming: American security in the 21st century. *Major Themes and Implications: The Phase I Report on the Emerging Global Security Environment for the First Quarter of the 21st Century. Washington, DC: US Commission on National Security/21st Century* (1999).

[92] HAWKINS, R., AND KELLY, T. A software safety argument pattern catalogue. *The University of York* (2013).

[93] HE, Q.-Y., HE, Q.-Z., DENG, X.-C., YAO, L., MENG, E., LIU, Z.-H., AND LIANG, S.-P. ATDB: a uni-database platform for animal toxins. *Nucleic acids research 36*, suppl_1 (2007), D293–D297.

[94] HENDERSON, D. A. Bioterrorism as a public health threat. *Emerging infectious diseases 4*, 3 (1998), 488.

[95] HERZIG, V., WOOD, D. L., NEWELL, F., CHAUMEIL, P.-A., KAAS, Q., BINFORD, G. J., NICHOLSON, G. M., GORSE, D., AND KING, G. F. ArachnoServer 2.0, an updated online resource for spider toxin sequences and structures. *Nucleic acids research 39*, suppl_1 (2010), D653–D657.

[96] HESSEL, A., GOODMAN, M., AND KOTLER, S. Hacking the president's DNA. *The Atlantic 310*, 4 (2012), 83.

[97] HITZ, M., AND MONTAZERI, B. Measuring coupling and cohesion in object-oriented systems. *University of Vienna* (1995).

[98] HUCKA, M., FINNEY, A., SAURO, H. M., BOLOURI, H., DOYLE, J. C., KITANO, H., ARKIN, A. P., BORNSTEIN, B. J., BRAY, D.,

CORNISH-BOWDEN, A., ET AL. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 4 (2003), 524–531.

[99] HUDSON, C. M. Taking local control of genomics machines through BWA. `coreymhudson.github.io/bwa_vulnerabilties/`, 2019.

[100] IGEM. BioBrick BBaK177029. `parts.igem.org/Part:BBa_K177029`, 2009.

[101] IGEM. iGEM 2018 Annual Report. `igem.org/wiki/images/d/d1/IGEM_2018_report.pdf`, 2018.

[102] IGEM. iGEM Main Page. `igem.org`, 2019.

[103] IGEM TEAM GHANA 2017. Team Ghana 2017 Hardware Page. `2017.igem.org/Team:AshesiGhana/Hardware`, 2017.

[104] IKEMOTO, L. C. DIY Bio: hacking life in biotech's backyard. *UCDL Rev. 51* (2017), 539.

[105] IMAI, M., WATANABE, T., HATTA, M., DAS, S. C., OZAWA, M., SHINYA, K., ZHONG, G., HANSON, A., KATSURA, H., WATANABE, S., ET AL. Experimental adaptation of an influenza H5 HA confers respiratory droplet transmission to a reassortant H5 HA/H1N1 virus in ferrets. *Nature 486*, 7403 (2012), 420–428.

[106] INTERNATIONAL COMMISSION ON RADIOLOGICAL PROTECTION. Main page. `icrp.org`, 2019.

[107] ISLAM, S. A., HEIL, B. J., KEARNEY, C. M., AND BAKER, E. J. Protein classification using modified n-grams and skip-grams. *Bioinformatics 34*, 9 (2018), 1481–1487.

[108] Jagadevan, S., Banerjee, A., Banerjee, C., Guria, C., Tiwari, R., Baweja, M., and Shukla, P. Recent developments in synthetic biology and metabolic engineering in microalgae towards biofuel production. *Biotechnology for biofuels 11*, 1 (2018), 185.

[109] Japan. Investigation framework to strengthen the biological weapons convention. `undocs.org/en/BWC/MSP/2018/MX.5/WP.1`, 2018.

[110] Jee, E., Lee, I., and Sokolsky, O. Assurance cases in model-driven development of the pacemaker software. In *Leveraging Applications of Formal Methods, Verification, and Validation* (Berlin, Heidelberg, 2010), T. Margaria and B. Steffen, Eds., Springer Berlin Heidelberg, pp. 343–356.

[111] Johnson, N., and Kelly, T. Devil's in the detail: Through-life safety and security co-assurance using SSAF. In *Computer Safety, Reliability, and Security* (2019), Springer International Publishing, pp. 299–314.

[112] Juarez Dominguez, A. L., Partridge, B. G., and Joyce, J. J. Creating safety assurance cases for rebreather systems. 34–39.

[113] Jungo, F., and Bairoch, A. Tox-Prot, the toxin protein annotation program of the Swiss-Prot protein knowledgebase. *Toxicon 45*, 3 (2005), 293–301.

[114] Kaas, Q., Westermann, J.-C., Halai, R., Wang, C. K., and Craik, D. J. ConoServer, a database for conopeptide sequences and structures. *Bioinformatics 24*, 3 (2007), 445–446.

[115] Kalupa, N. H. Black biology: Genetic engineering, the future of bioterrorism, and the need for greater international and community regulation of synthetic biology. *Wis. Int'l LJ 34* (2016), 952.

[116] KELLY, T. A systematic approach to safety case management. *SAE transactions* (2004), 257–266.

[117] KELLY, T., AND WEAVER, R. The goal structuring notation–a safety argument notation. *University of York* (2004), 6.

[118] KIS, Z., PEREIRA, H. S., HOMMA, T., PEDRIGI, R. M., AND KRAMS, R. Mammalian synthetic biology: emerging medical applications. *Journal of The Royal Society Interface 12*, 106 (2015), 20141000.

[119] KITADA, T., DIANDRETH, B., TEAGUE, B., AND WEISS, R. Programming gene and engineered-cell therapies with synthetic biology. *Science 359*, 6376 (2018), eaad1067.

[120] KNIGHT, T. Idempotent vector design for standard assembly of biobricks. *MIT Libraries* (2003).

[121] KOONIN, E. V. Does the central dogma still stand? *Biology direct 7*, 1 (2012), 27.

[122] KORDY, B., MAUW, S., RADOMIROVIĆ, S., AND SCHWEITZER, P. Foundations of attack–defense trees. In *International Workshop on Formal Aspects in Security and Trust* (2010), Springer, pp. 80–95.

[123] KORDY, B., MAUW, S., AND SCHWEITZER, P. Quantitative questions on attack–defense trees. In *International Conference on Information Security and Cryptology* (2012), Springer, pp. 49–64.

[124] KORNECKI, A., AND ZALEWSKI, J. Software certification for safety-critical systems: A status report. In *2008 International Multiconference on Computer Science and Information Technology* (2008), IEEE, pp. 665–672.

[125] KREAMER, D. K. The past, present, and future of water conflict and international security. *Journal of Contemporary Water Research & Education 149*, 1 (2012), 87–95.

[126] KRIEGER, L. M. Bay area biohacker tells you how to edit your DNA. does that make him a criminal? `mercurynews.com/2019/05/18/bay-area-biohacker-tells-you-how-to-edit-your-dna-does-that-make-him-a-crimi` 2019.

[127] KWOK, R. Five hard truths for synthetic biology. *Nature News 463*, 7279 (2010), 288–290.

[128] LANDRAIN, T., MEYER, M., PEREZ, A. M., AND SUSSAN, R. Do-it-yourself biology: challenges and promises for an open science and technology movement. *Systems and synthetic biology 7*, 3 (2013), 115–126.

[129] LANGARI, Z., AND MAIBAUM, T. Safety cases: A review of challenges. In *2013 1st International Workshop on Assurance Cases for Software-Intensive Systems (ASSURE)* (May 2013), pp. 1–6.

[130] LANGMEAD, B., TRAPNELL, C., POP, M., AND SALZBERG, S. L. Ultrafast and memory-efficient alignment of short dna sequences to the human genome. *Genome biology 10*, 3 (2009), R25.

[131] LEE, S. K., CHOU, H., HAM, T. S., LEE, T. S., AND KEASLING, J. D. Metabolic engineering of microorganisms for biofuels production: from bugs to synthetic biology to fuels. *Current opinion in biotechnology 19*, 6 (2008), 556–563.

[132] LEE, S. M. This guy says he's the first person to attempt editing his dna with CRISPR. `buzzfeednews.com/article/stephaniemlee/this-biohacker-wants-to-edit-his-own-dna#.evELlvD9p`, 2017.

[133] LEVESON, N. G. *Engineering a Safer World: Systems Thinking Applied to Safety*. MIT Press, 2012.

[134] LEYS, C., LEY, C., KLEIN, O., BERNARD, P., AND LICATA, L. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology 49*, 4 (2013), 764–766.

[135] LIGON, B. L. Plague: A review of its history and potential as a biological weapon. *Seminars in Pediatric Infectious Diseases 17*, 3 (2006), 161 – 170. Return of the Gram-Positives.

[136] LIN, C.-L., AND SHEN, W. Applying safety case pattern to generate assurance cases for safety-critical systems. 255–262.

[137] LIN, C.-L., AND SHEN, W. Generation of assurance cases for medical devices. 127–140.

[138] LITTLEWOOD, J. *The Biological Weapons Convention: A Failed Revolution*. Ashgate, 2005.

[139] LIU, B., WU, H., AND CHOU, K.-C. Pse-in-one 2.0: an improved package of web servers for generating various modes of pseudo components of dna, rna, and protein sequences. *Natural Science 9*, 04 (2017), 67.

[140] LIU, Z. Bioweapons . . . for dummies? `thebulletin.org/2015/09/bioweapons-for-dummies/`, 2015.

[141] LOCATELLI, A. The first injection in a human being of macromolecules whose primary structure was developed from a religious text. *OSF Preprints* (2018).

[142] Lotufo, R., She, S., Berger, T., Czarnecki, K., and Wasowski, A. Evolution of the linux kernel variability model. In *Proceedings of the 14th International Conference on Software Product Lines: Going Beyond* (2010), SPLC'10, pp. 136–150.

[143] Luo, X., Reiter, M. A., d'Espaux, L., Wong, J., Denby, C. M., Lechner, A., Zhang, Y., Grzybowski, A. T., Harth, S., Lin, W., et al. Complete biosynthesis of cannabinoids and their unnatural analogues in yeast. *Nature 567*, 7746 (2019), 123.

[144] Lusito, E., Felice, B., D'Ario, G., Ogier, A., Montani, F., Di Fiore, P. P., and Bianchi, F. Unraveling the role of low-frequency mutated genes in breast cancer. *Bioinformatics* (2018).

[145] Lutz, R. R., Lutz, J. H., Lathrop, J. I., Klinge, T. H., Mathur, D., Stull, D. M., Bergquist, T. G., and Henderson, E. R. Requirements analysis for a product family of DNA nanodevices. In *2012 20th IEEE International Requirements Engineering Conference* (Sep. 2012), pp. 211–220.

[146] Maksimov, M., Kokaly, S., and Chechik, M. A survey of tool-supported assurance case assessment techniques. *ACM Comput. Surv. 52*, 5 (2019), 101:1–101:34.

[147] Mallory, A. C., Reinhart, B. J., Jones-Rhoades, M. W., Tang, G., Zamore, P. D., Barton, M. K., and Bartel, D. P. MicroRNA control of PHABULOSA in leaf development: importance of pairing to the microRNA 5' region. *The EMBO journal 23*, 16 (2004), 3356–3364.

[148] Mandel, G. N., and Marchant, G. E. The living regulatory challenges of synthetic biology. *Iowa L. Rev. 100* (2014), 155.

[149] MANDELL, D. J., LAJOIE, M. J., MEE, M. T., TAKEUCHI, R., KUZNETSOV, G., NORVILLE, J. E., GREGG, C. J., STODDARD, B. L., AND CHURCH, G. M. Biocontainment of genetically modified organisms by synthetic protein design. *Nature 518*, 7537 (2015), 55–60.

[150] MAURER, S. M. End of the beginning or beginning of the end-synthetic biology's stalled security agenda and the prospects for restarting it. *Val. UL Rev. 45* (2010), 1387.

[151] MCLAUGHLIN, J. A., MYERS, C. J., ZUNDEL, Z., MISIRLI, G., ZHANG, M., OFITERU, I. D., GONI-MORENO, A., AND WIPAT, A. Synbiohub: a standards-enabled design repository for synthetic biology. *ACS synthetic biology 7*, 2 (2018), 682–688.

[152] MOLE, B. Part cow, part . . . bacterium? biotech company makes heifer of gene-editing blunder. `https://arstechnica.com/science/2019/09/part-cow-part-bacterium-biotech-company-makes-heifer-of-gene-editing-blunder` 2019.

[153] MOLTENI, M. Brazil's Plans for Gene-Edited Cows Got Scrapped–Here's Why. `https://www.wired.com/story/brazils-plans-for-gene-edited-cows-got-scrappedheres-why/`. Last accessed: January 26, 2020.

[154] MONTALVILLO, L., AND DÍAZ, O. Tuning GitHub for SPL development: Branching models & repository operations for product engineers. In *Proceedings of the 19th International Conference on Software Product Line* (New York, NY, USA, 2015), SPLC '15, ACM, pp. 111–120.

[155] MURCH, R. S., SO, W. K., BUCHHOLZ, W. G., RAMAN, S., AND PECCOUD, J. Cyberbiosecurity: an emerging new discipline to help safeguard the bioeconomy. *Frontiers in bioengineering and biotechnology 6* (2018), 39.

[156] NATIONAL ACADEMIES OF SCIENCES ENGINEERING AND MEDICINE AND OTHERS. *Biodefense in the Age of Synthetic Biology*. National Academies Press, 2018.

[157] NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. Cybersecurity framework. `nist.gov/cyberframework`, 2014.

[158] NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. NIST-FDA workshop report: Standards for pathogen detection via next generation sequencing. `nist.gov/mml/bbd/microbial-metrology/nistfda-workshop-standards-pathogen-detection-nextgeneration-sequencing`, 2015.

[159] NATIONAL SCIENCE ADVISORY BOARD FOR BIOSECURITY. NSABB Homepage. `osp.od.nih.gov/biotechnology/national-science-advisory-board-for-biosecurity-nsabb/`, 2018.

[160] NATURE EDITORIAL. Gene intelligence. `nature.com/news/gene-intelligence-1.19515`, 2016.

[161] NEGI, S. S., SCHEIN, C. H., LADICS, G. S., MIRSKY, H., CHANG, P., RASCLE, J.-B., KOUGH, J., STERCK, L., PAPINENI, S., JEZ, J. M., ET AL. Functional classification of protein toxins as a basis for bioinformatic screening. *Scientific reports 7*, 1 (2017), 13940.

[162] NEY, P., CEZE, L., AND KOHNO, T. Genotype extraction and false relative attacks: Security risks to third-party genetic genealogy services beyond identity inference.

[163] NEY, P., KOSCHER, K., ORGANICK, L., CEZE, L., AND KOHNO, T. Computer security, privacy, and DNA sequencing: Compromising computers with synthesized DNA, privacy leaks, and more. *26th USENIX Security Symposium.* (2017).

[164] NIELSEN, A. A., DER, B. S., SHIN, J., VAIDYANATHAN, P., PARALANOV, V., STRYCHALSKI, E. A., ROSS, D., DENSMORE, D., AND VOIGT, C. A. Genetic circuit design automation. *Science 352*, 6281 (2016), aac7341.

[165] NOYCE, R. S., LEDERMAN, S., AND EVANS, D. H. Construction of an infectious horsepox virus vaccine from chemically synthesized DNA fragments. *PLoS One 13*, 1 (2018), e0188453.

[166] ORMAN, H. The morris worm: A fifteen-year perspective. *IEEE Security & Privacy 99*, 5 (2003), 35–43.

[167] OSHANA, R. Software engineering of embedded and real-time systems. In *Software Engineering for Embedded Systems*. Elsevier, 2013, pp. 1–32.

[168] PADDON, C. J., AND KEASLING, J. D. Semi-synthetic artemisinin: a model for the use of synthetic biology in pharmaceutical development. *Nature Reviews Microbiology 12*, 5 (2014), 355.

[169] PAN, Y., WANG, S., ZHANG, Q., LU, Q., SU, D., ZUO, Y., AND YANG, L. Analysis and prediction of animal toxins by various chou's pseudo components and reduced amino acid compositions. *Journal of Theoretical Biology 462* (2019), 221 – 229.

[170] PECCOUD, J., ANDERSON, J. C., CHANDRAN, D., DENSMORE, D., GALDZICKI, M., LUX, M. W., RODRIGUEZ, C. A., STAN, G.-B., AND SAURO, H. M. Essential information for synthetic DNA sequences. *Nature Biotechnology 29*, 1 (2011), 22.

[171] PECCOUD, J., GALLEGOS, J. E., MURCH, R., BUCHHOLZ, W. G., AND RAMAN, S. Cyberbiosecurity: from naive trust to risk awareness. *Trends in biotechnology 36*, 1 (2018), 4–7.

[172] PECK, A. Re-framing biotechnology regulation. *Food & Drug LJ 72* (2017), 314.

[173] PLAKIDAS, K., STEVANETIC, S., SCHALL, D., IONESCU, T. B., AND ZDUN, U. How do software ecosystems evolve? a quantitative assessment of the r ecosystem. In *Proceedings of the 20th International Systems and Software Product Line Conference* (2016), SPLC '16, pp. 89–98.

[174] PRESIDENTIAL COMMISSION FOR THE STUDY OF BIOETHICAL ISSUES. New directions: The ethics of synthetic biology and emerging technologies. *DC: Presidential Commission for the Study of Bioethical Issues* (2010).

[175] QIU, J. Chinese government funding may have been used for 'CRISPR babies' project, documents suggest. `statnews.com/2019/02/25/crispr-babies-study-china-government-funding/`, 2019.

[176] QIU, J. Chinese government funding may have been used for 'CRISPR babies' project, documents suggest. `www.statnews.com/2019/02/25/crispr-babies-study-china-government-funding/`, 2019.

[177] QUAN, J., AND TIAN, J. Circular polymerase extension cloning of complex gene libraries and pathways. *PloS one 4*, 7 (2009), e6441.

[178] REEVES, R., VOENEKY, S., CAETANO-ANOLLÉS, D., BECK, F., AND BOËTE, C. Agricultural research, or a new bioweapon system? *Science 362*, 6410 (2018), 35–37.

[179] RICE, W. P. Medical device risk based evaluation and maintenance using fault tree analysis. *Biomedical Instrumentation & Technology 41*, 1 (2007), 76–82.

[180] RINCON, P. Gene editing is GM, says european court. `bbc.com/news/science-environment-44953100`, 2018.

[181] ROSSELLO, R. A., AND KOHN, D. H. Cell communication and tissue engineering. *Communicative & integrative biology 3*, 1 (2010), 53–56.

[182] ROTH, E. A guide to DIYbio. `medium.com/@ThatMrE/a-guide-to-diybio-updated-2019-abd0956cdf74`, 2019.

[183] RUIZ, A., HABLI, I., AND ESPINOZA, H. Towards a case-based reasoning approach for safety assurance reuse. In *Computer Safety, Reliability, and Security* (Berlin, Heidelberg, 2012), F. Ortmeier and P. Daniel, Eds., Springer Berlin Heidelberg, pp. 22–35.

[184] RUSHBY, J. The interpretation and evaluation of assurance cases. *Comp. Science Laboratory, SRI International, Tech. Rep. SRI-CSL-15-01* (2015).

[185] SAHA, S., AND RAGHAVA, G. P. BTXpred: prediction of bacterial toxins. *In silico biology 7*, 4, 5 (2007), 405–412.

[186] SAHA, S., AND RAGHAVA, G. P. Prediction of neurotoxins based on their function and source. *In silico biology 7*, 4, 5 (2007), 369–387.

[187] SALTER, C., SAYDJARI, O. S., SCHNEIER, B., AND WALLNER, J. Toward a secure system engineering methodology. In *Proceedings of the 1998 workshop on New security paradigms* (1998), ACM, pp. 2–10.

[188] SCHMIDT, M. Diffusion of synthetic biology: a challenge to biosafety. *Systems and synthetic biology 2*, 1-2 (2008), 1–6.

[189] SCHUSTER, S. C. Next-generation sequencing transforms today's biology. *Nature methods 5*, 1 (2007), 16.

[190] SCHYFTER, P., FROW, E., AND CALVERT, J. Synthetic biology: making biology into an engineering discipline. *Engineering Studies 5*, 1 (2013), 1–5.

[191] SELGELID, M. J., AND WEIR, L. Reflections on the synthetic production of poliovirus. *Bulletin of the Atomic Scientists 66*, 3 (2010), 1–9.

[192] SHANNON, C. E. A mathematical theory of communication. *Bell system technical journal 27*, 3 (1948), 379–423.

[193] SHOSTACK, A. *Threat modeling: Designing for security*. John Wiley & Sons, 2014.

[194] SIDHARTH, B. G. Black-hole thermodynamics and electromagnetism. *Foundations of Physics Letters 19*, 1 (2006), 87–94.

[195] SINCERO, J., SCHIRMEIER, H., SCHRÖDER-PREIKSCHAT, W., AND SPINCZYK, O. Is the linux kernel a software product line. In *Proc. SPLC Workshop on Open Source Software and Product Lines* (2007).

[196] STIRLING, F., BITZAN, L., O?KEEFE, S., REDFIELD, E., OLIVER, J. W., WAY, J., AND SILVER, P. A. Rational design of evolutionarily stable microbial kill switches. *Molecular Cell 68*, 4 (2017), 686 – 697.e3.

[197] SUJAN, M. A., KOORNNEEF, F., CHOZOS, N., POZZI, S., AND KELLY, T. Safety cases for medical devices and health information technology: involving health-care organisations in the assurance of safety. *Health informatics journal 19*, 3 (2013), 165–182.

[198] SWANSON, J., COHEN, M. B., DWYER, M. B., GARVIN, B. J., AND FIRESTONE, J. Beyond the rainbow: self-adaptive failure avoidance in configurable systems. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering* (2014), ACM, pp. 377–388.

[199] SYNTHETIC BIOLOGY:SB2.0. Biosecurity resolutions. `openwetware.org/wiki/`, 2006.

[200] SZCZYGIELSKA, M., AND JARZĘBOWICZ, A. Assurance case patterns on-line catalogue. In *Advances in Dependability Engineering of Complex Systems* (Cham, 2018), W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak, and J. Kacprzyk, Eds., Springer International Publishing, pp. 407–417.

[201] TAVELLA, F., GIARETTA, A., DOOLEY-CULLINANE, T. M., CONTI, M., COFFEY, L., AND BALASUBRAMANIAM, S. DNA molecular storage system: Transferring digitally encoded information through bacterial nanonetworks. *arXiv preprint arXiv:1801.04774* (2018).

[202] THE ODIN. Genetic engineering home lab kit. `the-odin.com/genetic-engineering-home-lab-kit/`, 2019.

[203] THERMO FISHER SCIENTIFIC. Geneart gene synthesis kit. `https://web.archive.org/web/20180826234509/http:/www.thermofisher.com/order/catalog/product/A14971`, 2019.

[204] THOMPSON, E. M., BROWN, M., DOBRIKOVA, E., RAMASWAMY, V., TAYLOR, M. D., MCLENDON, R., SANKS, J., CHANDRAMOHAN, V., BIGNER, D., AND GROMEIER, M. Poliovirus receptor (cd155) expression in pediatric brain tumors mediates oncolysis of medulloblastoma and pleomorphic

xanthoastrocytoma. *Journal of Neuropathology & Experimental Neurology 77*, 8 (2018), 696–702.

[205] THÜM, T., APEL, S., KÄSTNER, C., SCHAEFER, I., AND SAAKE, G. A classification and survey of analysis strategies for software product lines. *ACM Comput. Surv. 47*, 1 (June 2014), 6:1–6:45.

[206] TRUMP, B., CUMMINGS, C., KUZMA, J., AND LINKOV, I. A decision analytic model to guide early-stage government regulatory action: Applications for synthetic biology. *Regulation & Governance 12*, 1 (2018), 88–100.

[207] TRUMP, B. D. Synthetic biology regulation and governance: Lessons from TAPIC for the united states, european union, and singapore. *Health Policy 121*, 11 (2017), 1139–1146.

[208] TUCKER, J. B., AND ZILINSKAS, R. A. The promise and perils of synthetic biology. *The New Atlantis*, 12 (2006), 25–45.

[209] TUMPEY, T. M., BASLER, C. F., AGUILAR, P. V., ZENG, H., SOLÓRZANO, A., SWAYNE, D. E., COX, N. J., KATZ, J. M., TAUBENBERGER, J. K., PALESE, P., ET AL. Characterization of the reconstructed 1918 spanish influenza pandemic virus. *Science 310*, 5745 (2005), 77–80.

[210] UNITED NATIONS OFFICE FOR DISARMAMENT AFFAIRS. The biological weapons convention, an introduction. `un.org/disarmament/publications/more/the-biological-weapons-convention-an-introduction/`, 2019.

[211] UNITED STATES OF AMERICA. Institutional strengthening of the BWC. `undocs.org/en/BWC/MSP/2018/MX.5/WP.3`, 2018.

[212] USDA. Bioengineered foods disclosure. `ams.usda.gov/rules-regulations/be`, 2019.

[213] USDA. National bioengineered food disclosure standard. federalregister.gov/documents/2018/12/21/2018-27283/national-bioengineered-food-disclosure-standard, 2019.

[214] VALVERDE, S., PORCAR, M., PERETÓ, J., AND SOLÉ, R. V. The software crisis of synthetic biology. *bioRxiv* (2016).

[215] VALVERDE, S., PORCAR, M., PERETO, J., AND SOLE, R. V. The software crisis of synthetic biology. *bioRxiv* (2016), 041640.

[216] VAN DER LINDEN, F. J., SCHMID, K., AND ROMMES, E. *Software product lines in action: the best industrial practice in product line engineering*. Springer Science & Business Media, 2007.

[217] WANG, F., AND ZHANG, W. Synthetic biology: Recent progress, biosafety and biosecurity concerns, and possible solutions. *Journal of Biosafety and Biosecurity* (2019).

[218] WARRICK, J. Papers stolen in a daring israeli raid on tehran archive reveal the extent of iran's past weapons research. washingtonpost.com/world/national-security/papers-stolen-in-a-daring-israeli-raid-on-tehran-archive-reveal-the/extent-of-irans-past-weapons-research/2018/07/15/0f7911c8-877c-11e8-8553-a3ce89036c78_story.html, 2018.

[219] WATANABE, T., ZHONG, G., RUSSELL, C. A., NAKAJIMA, N., HATTA, M., HANSON, A., MCBRIDE, R., BURKE, D. F., TAKAHASHI, K., FUKUYAMA, S., ET AL. Circulating avian influenza viruses closely related to the 1918 virus have pandemic potential. *Cell host & microbe 15*, 6 (2014), 692–705.

[220] WATERS, C. M., AND BASSLER, B. L. Quorum sensing: cell-to-cell communication in bacteria. *Annu. Rev. Cell Dev. Biol. 21* (2005), 319–346.

[221] WEBER, W., AND FUSSENEGGER, M. Emerging biomedical applications of synthetic biology. *Nature Reviews Genetics 13*, 1 (2012), 21.

[222] WEBER, W., STELLING, J., RIMANN, M., KELLER, B., DAOUD-EL BABA, M., WEBER, C. C., AUBEL, D., AND FUSSENEGGER, M. A synthetic time-delay circuit in mammalian cells and mice. *Proceedings of the National Academy of Sciences of the United States of America 104* (2007), 2643–2648.

[223] WEINSTOCK, C. B., AND GOODENOUGH, J. B. Towards an assurance case practice for medical devices. *Carnegie Mellon University* (2009).

[224] WETZEL, V., BRAULT, V., AND VARRELMANN, M. Production of a beet chlorosis virus full-length cDNA clone by means of gibson assembly and analysis of biological properties. *Journal of General Virology 99*, 11 (2018), 1522–1527.

[225] WHEELIS, M., CASAGRANDE, R., AND MADDEN, L. V. Biological attack on agriculture: Low-tech, high-impact bioterrorism: Because bioterrorist attack requires relatively little specialized expertise and technology, it is a serious threat to US agriculture and can have very large economic repercussions. *BioScience 52*, 7 (2002), 569–576.

[226] WHITAKER, W. B., SANDOVAL, N. R., BENNETT, R. K., FAST, A. G., AND PAPOUTSAKIS, E. T. Synthetic methylotrophy: engineering the production of biofuels and chemicals based on the biology of aerobic methanol utilization. *Current opinion in biotechnology 33* (2015), 165–175.

[227] WIKIPEDIA. Gilbert U-238 Atomic Energy Laboratory. `en.wikipedia.org/wiki/Gilbert_U-238_Atomic_Energy_Laboratory`, 2018.

[228] WIKIPEDIA. Sea Monkeys. `en.wikipedia.org/wiki/Sea-Monkeys`, 2019.

[229] WINFREE, E. On the computational power of DNA annealing and ligation. In *DNA Based Computers* (1995).

[230] WOODS, D., DOTY, D., MYHRVOLD, C., HUI, J., ZHOU, F., YIN, P., AND WINFREE, E. Diverse and robust molecular algorithms using reprogrammable dna self-assembly. *Nature 567* (2019), 366–372.

[231] WRIGHT, R. C., AND NEMHAUSER, J. Plant synthetic biology: Quantifying the "known unknowns" and discovering the "unknown unknowns". *Plant Physiology 179*, 3 (2019), 885–893.

[232] YANG, J., YU, S., GONG, B., AN, N., AND ALTEROVITZ, G. Biobrick chain recommendations for genetic circuit design. *Computers in biology and medicine 86* (2017), 31–39.

[233] YANG, X., CHOCKALINGAM, S. P., AND ALURU, S. A survey of error-correction methods for next-generation sequencing. *Briefings in bioinformatics 14*, 1 (2012), 56–66.

[234] YASSERI, S. Evidence-based practice in subsea engineering. *Underwater Technology 32*, 4 (2015), 231–244.

[235] YOU, E. Safeguarding the bioeconomy: U.S. opportunities and challenges. `www.ehidc.org/resources/safeguarding-bioeconomy-us-opportunities-and-challenges`, 2017.

[236] YU, W., YUAN, C.-A., QIN, X., HUANG, Z.-K., AND SHANG, L. Hierarchical attention network for predicting dna-protein binding sites. In *International Conference on Intelligent Computing* (2019), Springer, pp. 366–373.

[237] YUKI, H., HOUGH, L., SAGEMAN, M., DANZIG, R., KOTANI, R., AND LEIGHTON, T. Aum shinrikyo: Insights into how terrorists develop biological and chemical weapons. cnas.org/publications/reports/aum-shinrikyo-second-edition-english, 2011.

[238] ZAYNER, J. Our statement about beer brewing using our yeast kits. the-odin.com/blog/our-statement-about-beer-brewing-using-our-yeast-kits/, 2016.

[239] ZENG, W., WU, M., AND JIANG, R. Prediction of enhancer-promoter interactions via natural language processing. *BMC genomics 19*, 2 (2018), 84.

[240] ZHANG, S. Whatever happened to the glowing plant kickstarter? theatlantic.com/science/archive/2017/04/whatever-happened-to-the-glowing-plant-kickstarter/523551/, 2017.

[241] ZHANG, S. A biohacker regrets publicly injecting himself with CRISPR. theatlantic.com/science/archive/2018/02/biohacking-stunts-crispr/553511/, 2018.

[242] ZILINSKAS, R. A., AND MAUGER, P. Biosecurity in putin's russia. *Notes 25* (2018), 33.

[243] unclemilton.com/ant_farm/ant_farm/.

[244] α-Latrotoxin-Lt1a. arachnoserver.org/toxincard.html?id=60.

[245] https://blast.ncbi.nlm.nih.gov/Blast.cgi.

[246] https://ebrc.org/what-is-synbio/.

[247] geneuniversal.com.

[248] genscript.com.

[249] https://2019.igem.org/About.

[250] https://unl-igem-test.netlify.com/.

[251] 2015.igem.org/Team:BIT-China/Safety.

[252] parts.igem.org/cgi/partsdb/pgroup.cgi?pgroup=Coding.

[253] parts.igem.org/Cell_death.

[254] 2016.igem.org/Team:NCTU_Formosa/Safety.

[255] 2017.igem.org/Human_Practices.

[256] 2017.igem.org/Team:IONIS-PARIS/applied-design/risk-avoidance.

[257] parts.igem.org/Special:Search?search=kill+switch.

[258] parts.igem.org/wiki/index.php?title=Special%3ASearch&profile=default&search=kill-switch&fulltext=Search.

[259] igem.org/Main_Page.

[260] 2015.igem.org/Team:Manchester-Graz/Safety.

[261] 2015.igem.org/Team:UMaryland/HokSok.

[262] 2017.igem.org/Team:UNebraska-Lincoln/Safety.

[263] parts.igem.org/cgi/partsdb/pgroup.cgi?pgroup=Regulatory.

[264] parts.igem.org/cgi/partsdb/pgroup.cgi?pgroup=RBS.

[265] http://parts.igem.org.

[266] http://parts.igem.org/Biosafety.

[267] 2015.igem.org/Safety/Do_Not_Release.

[268] https://2019.igem.org/Safety/Risk_Groups.

[269] https://igem.org/Safety.

[270] 2015.igem.org/Team:SJTU-BioX-Shanghai/Safety.

[271] 2015.igem.org/Team:SZU_China/Safety.

[272] parts.igem.org/cgi/partsdb/pgroup.cgi?pgroup=terminator.

[273] 2015.igem.org/Team:Tsinghua/Safety.

[274] 2015.igem.org/Team:Uppsala/Safety.

[275] 2016.igem.org/Team:Wageningen_UR/Safety.

[276] 2015.igem.org/Team:ZJU-China/Safety.

[277] 2016.igem.org/Team:ETH_Zurich/Safety.

[278] osp.od.nih.gov/wp-content/uploads/NIH_Guidelines.html.

[279] Pse-in-One 2.0. http://bioinformatics.hitsz.edu.cn/Pse-in-One2.0/DNA/Kmer/. Last accessed: April 22, 2020.

[280] sbolstandard.org.

[281] twistbioscience.com.

[282] UNL 2017 iGEM Team Page. http://2017.igem.org/Team:UNebraska-Lincoln. Last accessed: January 26, 2020.

[283] https://commons.wikimedia.org/wiki/File:Aminoacids_table.svg.