2021

Minimizing Reaction Systems

Matthew R. Thomas
University of North Florida, n01375183@unf.edu

Minimizing Reaction Systems

Matthew Thomas

A thesis submitted to the Department of Mathematics and Statistics

in partial fulfillment of the requirements for the degree of

Master of Science in Mathematical Sciences

UNIVERSITY OF NORTH FLORIDA

COLLEGE OF ARTS AND SCIENCES

April 29, 2021

Acknowledgments

I would like to thank the thesis committee chair Dr. Daniela Genova and committee members Dr. Michelle DeDeo and Dr. Zornitza Prodanoff.

Contents

Abstract

The theoretical model for reaction systems is a relatively new framework originally proposed as a mathematical model for biochemical processes which take place in living cells. Growing interest in this research area has lead to the abstraction of the model for non-biological purpose as well. Reaction systems, with a well understood behavior, have become important for studying transition systems. As with any mathematical model, we want to simplify a given implementation of the model as much as possible while maintaining functional equivalence. This paper discusses the formal model for reaction systems, how we can simplify them with minimization techniques, some of their capabilities and properties, and a comparison of those properties for minimal and non-minimal reaction systems. Original software written for the purpose of exploring reaction systems for this paper as well as well-known logic minimization algorithms instrumental in simplifying reaction systems are discussed.

1. Introduction

The theoretical model of reaction systems is a relatively new framework originally proposed in 2007 by Ehrenfeucht and Rozenberg (see [8]) as a mathematical model for biochemical processes which take place in living cells. Additionally, the model provides for the interaction between a reaction system and the larger environment containing it or between reaction system processes. These processes involve reactions which are based on the premise that the two main mechanisms of reactions are facilitation and inhibition: reactions occur if all of the facilitating reactants are present and none of the inhibiting entities are present.

Growing interest in this research area has lead to the abstraction of the theoretical model for non-biological purposes. Reaction systems' well-known behavior and their generative capacity for long sequences makes them particularly useful for studying state transition systems.

A topic of importance in studying any mathematical structure is that of equivalence and minimization. Reaction systems can easily be converted to Boolean vector functions in disjunctive normal form and vice versa. Thus, a functionally equivalent minimized reaction system can be obtained using logic formula minimization algorithms.

The paper is organized as follows.

In Chapter 2, the theoretical model of reaction systems is defined. Also discussed are functional equivalence and classes of reaction systems which are vital concepts throughout the paper.

In Chapter 3, more topics on functions defined by reaction systems are discussed. Context independent and interactive processes are defined and demonstrated for their capabilities to generate sequences. Some known facts are presented and proved about the lower bounds for lengths of sequences that can be generated by certain classes of reaction systems. Representing reaction systems as a set of Boolean functions is discussed in detail, and this topic

is instrumental in minimizing reaction systems.

In Chapter 4, subadditive properties of reaction system functions are defined as are special classes of minimal reaction systems. Characterization theorems are proved with an emphasis on the existence of functionally equivalent reaction system functions for the smallest class of reaction systems. Some of their properties are compared with larger classes of reaction systems.

Chapter 5 considers a way to minimize reaction systems using logic minimization algorithms. Provided is an elaborate example of finding a minimized reaction system which models a state transition system given a one-out directed graph representing that system.

Chapter 6 is a review of RS Tools which is original software written for the purpose of exploring reaction systems to provide insight into the topics discussed herein. Interspersed throughout the paper are references to RS Tools and how it was used in some of the examples.

2. Reactions and reaction systems

2.1 Reactions

In a living cell, biochemical reactions take place that result in the production of certain biochemical products when all the necessary reactants are present and none of the inhibiting entities are present to suppress the formation of the products. If all the necessary reactants are present but at least one of the inhibitors is present then no products result from the reaction. This process is mathematically formalized in the following definition which was introduced in [8].

Definition 2.1. A *reaction* is a 3-tuple $a = (R_a, I_a, P_a)$ of finite sets. If S is a set such that $R_a, I_a, P_a \subseteq S$ then we say that a is a *reaction in* S. The set R_a is the *reactant set* of a; the set I_a is the *inhibitor set* of a; and the set P_a is the *product set* of a. The reaction $(\varnothing, \varnothing, \varnothing)$ is called the *empty reaction* and denoted by Φ.

The reactant, inhibitor, and product sets may be denoted R, I, and P when the reaction they are associated with is understood. Sometimes we are interested in $R_a \cup I_a$ which we will denote as M_a and which we will refer to as the resources of a.

When the conditions are present for a reaction to occur, the reaction transforms the reactants to produce its set of products. In other words, for a reaction a and a finite set $T \subseteq S$, the *result of* a on T, denoted $res_a(T)$, is defined in [8] by:

$$res_a(T) = \begin{cases} P_a, & \text{if } R_a \subseteq T \text{ and } I_a \cap T = \varnothing \\ \varnothing, & \text{otherwise.} \end{cases}$$

When the conditions for the reaction to occur are present, we say that a is *enabled on* T, denoted $en_a(T)$, otherwise a is not enabled on T.

Example 2.1.1. Suppose reaction $a = (\{a, b\}, \{c, d\}, \{a, c\})$ can potentially occur in an environment which contains a subset of the background set $S = \{a, b, c, d, e\}$. For a selection of subsets $T_i \subseteq S$, the following shows the behavior of reaction a.

T	$en_a(T)$	$res_a(T)$
$\{a, b, e\}$	yes	$\{a, c\}$
$\{a, b, c\}$	no	$\varnothing$
$\{a, b\}$	yes	$\{a, c\}$
$\{a\}$	no	$\varnothing$

Example 2.1.2. In *E. coli* cells, the metabolism of lactose is controlled by the expression of a specific set of genes in a complex genetic regulatory system. The reaction system which models this regulatory system is discussed in [4]. One of the many reactions in this system determines whether or not the gene is expressed, allowing lactose to be digested. The biochemical environment where the reaction occurs involves the following entities representing various molecules: *lac* (for lactose), *cAMP–CAP*, *I-OP*, *Z*, *Y*, and *A*. If the reactants *lac* and *cAMP–CAP* are present and the inhibitor *I-OP* is not, then *Z*, *Y*, and *A* are produced allowing the expression of genes that cause the lactose to be digested. To model this reaction, let $S = \{$*lac, cAMP–CAP, I-OP, Z, Y, A*$\}$ and let $a = (\{$*lac, cAMP–CAP*$\}, \{$*I–OP*$\}, \{$*Z, Y, A*$\})$. Then for $T \subseteq S$, if $en_a(T)$ then $res_a(T) = \{$*Z, Y, A*$\}$ and lactose is digested, otherwise $res_a(T) = \varnothing$.

We often refers to sets of reactions. The set of all reactions over background set S is denoted by $rac(S)$. For a set $A \subseteq rac(S)$, the following are defined: $R_A = \bigcup_{a \in A} R_a$, $I_A = \bigcup_{a \in A} I_a$, and $P_A = \bigcup_{a \in A} P_a$.

Just as we have a result function for each individual reaction, we define the result of an entire set of reactions A on T by $res_A(T) = \bigcup \{res_a(T) \mid a \in A\}$. If $R_a \subseteq T$ and $I_a \cap T = \varnothing$, i.e. $en_a(T)$, for at least one $a \in A$, then we say that A is *enabled on* T, indicated by $en_A(T)$, otherwise we say that A is *not enabled on* T. If every reaction $a \in A$ is enabled on T, we say that A is *enabled by* T.

2.2 Reaction Systems

The previous section discussed individual reactions and sets of reactions. We can now define a reaction system, which originated in [8].

Definition 2.2. A *reaction system*, abbreviated *RS*, is an ordered pair $\mathcal{A} = (S, A)$ such that S is a finite set and $A \subseteq rac(S)$.

The elements of S are referred to as *entities*. The entities used as reactants and inhibitors are called *resources*. The set S, called the *background set of* $\mathcal{A}$, consists of all the entities used as the resources and products of the reactions of A.

For a reaction system $\mathcal{A} = (S, A)$ and a set $T \subseteq S$, the *result* of $\mathcal{A}$ on T (also referred to as the result function), denoted $res_{\mathcal{A}}(T)$ which is defined as the result of the reaction set A (i.e., $res_{\mathcal{A}}(T) = res_A(T) = \bigcup \{res_a(T) \mid a \in A\}$). The set $\{a \in A \mid a \text{ is enabled by } T\}$ is called the *T-activity* of $\mathcal{A}$ (or activity of $\mathcal{A}$ on T), denoted by $en_{\mathcal{A}}(T)$. The set T describes the *state* of the RS at any given moment. It is the set of entities that are present at that moment. Once the set of reactions A operates on the available resources specified by T, the state could change which would constitute a process which may also produce a sequence. Processes are discussed in section 3.1.

As defined above, $en_{\mathcal{A}}(T)$ is a set of reactions. In this paper, as in some others such as [5], the notation $en_a(T), en_A(T)$, and $en_{\mathcal{A}}(T)$ can also be understood, depending on the context, to mean "reaction a (resp. reaction set A, RS $\mathcal{A}$) is enabled on T".

2.3 Model assumptions

The reaction system model established in [8] includes a discussion about the following model assumptions which are believed to hold for the majority of biological reactions, and so they are axiomatic for the basic model.

2.3.1 Threshold assumption

In Example 2.1.2, an actual biological process is described which involves the presence of lactose (or lack thereof) in a living cell. There is nothing in the example that concerns the quantity of lactose molecules; we are only concerned about whether or not it is present. The threshold assumption of reaction systems is that if a resource is included in $T \in S$, for the purpose of evaluating $res(T)$, there is enough of it present for all enabled reactions in the system to consume that resource. According to [5], we can also assume that the concentration of resources present is irrelevant. This assumptions is also described as the *threshold nature of resources* [5], or in other words there is a *threshold supply* of resources [8].

2.3.2 Non-permanency assumption

In Example 2.1.2, a set of reactions take place in an environment which may contain reactants and inhibitors. For the model of reaction systems, we assume that once the set of reactions transforms the available reactants into products, the environment only consists of the products. For example, if lactose was present and needed to enable any of the reactions, those enabled reactions transform lactose into products, and lactose remains only if it is in the product set of the enabled reactions. Furthermore, all resources that were present but not include in the product set cease to exist. This describes the *non-permanency* assumption of the model.

2.3.3 Reactions are primary

Because of the non-permanency assumption, once the enabled reactions of a reaction system have converted reactants into products, a new state, $res_{\mathcal{A}}$, is created and the new environment is based only on the entities of the new state. As described in [8], reactions are primary while structures are secondary. Reaction do not transform states, they create states and no structure irrelevant to the enabled reactions is carried over. According to [8], this aspect of reaction systems is different from traditional models.

2.4 Classes of reaction systems

Classes of reaction systems (and RS functions which are discussed in the next chapter) are defined based on the cardinalities of the reactant and inhibitor sets of the corresponding reaction systems. A reaction system $\mathcal{A} = (R, I, P)$ is referred to as an (r, i) system if for every $a \in rac(\mathcal{A})$, $|R_a| \leq r$ and $|I_a| \leq i$ [15], [18]. In Example 3.2.1, the RS $\mathcal{A}$ is a $(1, 2)$ RS and $\mathcal{B}$ is a $(2, 1)$ RS. Examining the properties of $(1, 1)$ reaction systems is a focus of this paper, and simplifying reaction systems as much as possible to the extent of $(1, 1)$ systems is another focus.

2.5 Functional equivalence

As mentioned above, res is a function. Two reaction systems $\mathcal{A}$ and $\mathcal{B}$ over the same background set can be very different, e.g. they can be of different classes, they can have different cardinalities of reactions, and their reactions can be different, and yet the functions defined by them can be equivalent.

Definition 2.3. For two reaction systems, $\mathcal{A} = (S, A)$ and $\mathcal{B} = (S, B)$ where A is not necessarily identical to B, if $res_{\mathcal{A}}(T) = res_{\mathcal{B}}(T)$ for every $T \subseteq S$, then $\mathcal{A}$ is *functionally equivalent* to $\mathcal{B}$, denoted $\mathcal{A} \sim \mathcal{B}$. [8]

3. Processes, sequences, and functions

3.1 Interactive and context-independent processes

A reaction system process is a state transition process in which the result set of one step of the process then becomes the input resources for the next step of the process. If external input is added to the resources at any step, this can be viewed as interaction with the reaction system. These processes are defined in [8].

3.1.1 Context-independent process

Consider a RS $\mathcal{A} = (S, A)$ with $W_0 \subset S$. Let $D_1 = res_{\mathcal{A}}(W_0)$. Then let $D_2 = res_{\mathcal{A}}(D_1)$, $D_3 = res_{\mathcal{A}}(D_2)$, and so on. This process generates the sequence $\{D_i\}, i = 1, 2, 3, ...$

For a specific step in the sequence, say D_3, another way to view how D_3 is generated is: $D_3 = res_{\mathcal{A}}(res_{\mathcal{A}}(res_{\mathcal{A}}(W_0))) = res^3_{\mathcal{A}}(W_0)$. Except for the initial state W_0, there is no other context surrounding this process, i.e. there is no interaction. Hence, this process is called a *context-independent process*.

As an example, if $S = \{a, b, c\}$ and $A = \{(\{a, b\}, \{c\}, \{a, c\}), (\{a, c\}, \{b\}, \{a, b\})\}$, then the steps of the sequence for the initial state $W_0 = \{a, b\}$ are $\{a, c\}, \{a, b\}, \{a, c\}, \{a, b\}, ...,$ and we see that $res^i_{\mathcal{A}}(W_0) = \{a, c\}$ if i is even.

A much larger example of a context-independent reaction system function which, given a specific initial state, generates the first few terms of the Fibonacci sequence in a six-step process, is shown in Appendix A.3.

3.1.2 Interactive process

In the theoretical model for reaction systems, it is possible for additional entities from a source of interaction to be added to the result set of one step to be used as the resources for the next step. This process was originally defined in [8].

Definition 3.1. Let $\mathcal{A} = (S, A)$ be a RS. An *interactive process* π in $\mathcal{A}$ is a pair of finite sequences $\pi = (\gamma, \delta)$ such that, for some $n \geq 1$, $\gamma = C_0, C_1, ..., C_n$, $\delta = D_1, ..., D_n$ where $C_0, C_1, ..., C_n, D_1, ..., D_n \subseteq S$, $D_1 = res_{\mathcal{A}}(C_0)$ and, for each $1 \leq i \leq n$, $D_i = res_{\mathcal{A}}(C_{i-1} \cup D_{i-1})$. The sequence $C_0, C_1, ..., C_n$ is the *interaction sequence* (also referred to as *context sequence*) of π, and the sequence $D_1, ..., D_{n+1}$ is the *result sequence* of π. For each $1 \leq i \leq n$, we define $W_i = C_i \bigcup D_i$ and $W_0 = C_0$. The sequence $W_0, W_1, W_2, ..., W_n$ is called the sequence of *states of* π; while W_0 is called the initial state of π. For each $0 \leq j \leq n$, the set C_j is called the *context of* W_j. The sequence $E_0, E_1, ..., E_n$ of subsets of$A$ such that $E_i = en_{\mathcal{A}}(W_i)$, for all $1 \leq i \leq n-1$, is called the *activity sequence of* π.

Table 3.1: General interactive process

Step i	C_i	W_i	E_i	D_{i+1}
0	C_0	$W_0 = C_0$	$E_0 = en_{\mathcal{A}}(C_0)$	$D_1 = res_{\mathcal{A}}(C_0)$
1	C_1	$W_1 = C_1 \cup D_1$	$E_1 = en_{\mathcal{A}}(W_1)$	$D_2 = res_{\mathcal{A}}(W_1)$
2	C_2	$W_2 = C_2 \cup D_2$	$E_2 = en_{\mathcal{A}}(W_2)$	$D_3 = res_{\mathcal{A}}(W_3)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	C_n	$W_n = C_n \cup D_n$	$E_n = en_{\mathcal{A}}(W_n)$	$D_n = res_{\mathcal{A}}(W_n)$

Table 3.1 shows the general process that follow from Definition 3.1. In this table, $\gamma = \{C_i\}$ and $\delta = \{D_{i+1}\}$.

Example 3.1.1. In an interactive process, the reactions shown in Table 3.2 generate a sequence which oscillates between 0 and 1 until a trigger is applied (the interaction), and it switches to oscillating between 0 and -1. If the trigger is applied again, the RS switches to oscillating between 0 and 1 again. The trigger is applied through the context sequence.

Table 3.2: Reactions for Example 3.1.1

$a_1 = (\{0, h\}, \{tr\}, \{1\})$ $\quad$ $a_5 = (\{0\}, \{h, tr\}, \{-1\})$
$a_2 = (\{0, h, tr\}, \{1\}, \{-1\})$ $\quad$ $a_6 = (\{0, tr\}, \{h\}, \{1\})$
$a_3 = (\{1\}, \{tr\}, \{0, h\})$ $\quad$ $a_7 = (\{-1\}, \{tr\}, \{0\})$
$a_4 = (\{1, tr\}, \{0\}, \{0\})$ $\quad$ $a_8 = (\{-1, tr\}, \{0\}, \{0, h\})$

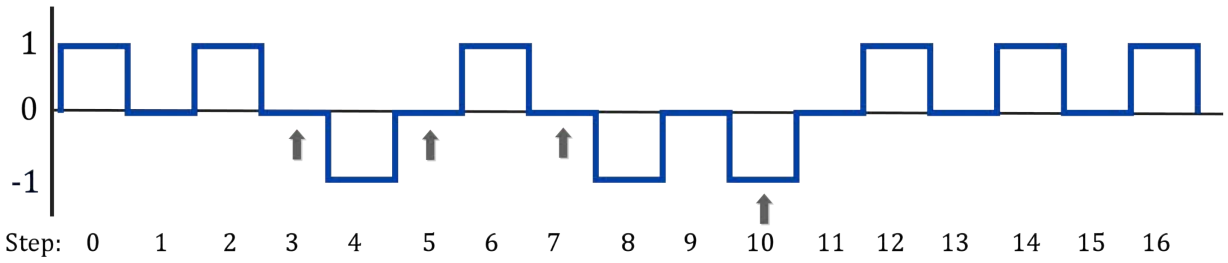

Figure 3.1: Visual representation of interactive process with triggers at arrows.

Table 3.3: Interactive process for Example 3.1.1.

Step i	C_i	W_i	E_i	D_{i+1}	Step i	C_i	W_i	E_i	D_{i+1}
0	$\{1\}$	$\{1\}$	$\{a_3\}$	$\{0, h\}$	8	$\{\varnothing\}$	$\{-1\}$	$\{a_7\}$	$\{0\}$
1	$\{\varnothing\}$	$\{0, h\}$	$\{a_1\}$	$\{1\}$	9	$\{\varnothing\}$	$\{0\}$	$\{a_5\}$	$\{-1\}$
2	$\{\varnothing\}$	$\{1\}$	$\{a_3\}$	$\{0, h\}$	10	$\{tr\}$	$\{-1, tr\}$	$\{a_8\}$	$\{0, h\}$
3	$\{tr\}$	$\{0, h, tr\}$	$\{a_2\}$	$\{-1\}$	11	$\{\varnothing\}$	$\{0, h\}$	$\{a_1\}$	$\{1\}$
4	$\{\varnothing\}$	$\{-1\}$	$\{a_7\}$	$\{0\}$	12	$\{\varnothing\}$	$\{1\}$	$\{a_3\}$	$\{0, h\}$
5	$\{tr\}$	$\{0, tr\}$	$\{a_8\}$	$\{1\}$	13	$\{\varnothing\}$	$\{0, h\}$	$\{a_1\}$	$\{1\}$
6	$\{\varnothing\}$	$\{1\}$	$\{a_3\}$	$\{0, h\}$	14	$\{\varnothing\}$	$\{1\}$	$\{a_3\}$	$\{0, h\}$
7	$\{tr\}$	$\{0, h, tr\}$	$\{a_2\}$	$\{-1\}$	15	$\{\varnothing\}$	$\{0, h\}$	$\{a_1\}$	$\{1\}$ (state 16)

Each step of the result sequence can be interpreted as 1, 0, or -1 if those entities are in the result set; there are sometimes additional entities in the result set which are needed for the RS to work. Suppose we want the result sequence to contain no extra entities (i.e., h), we can use the result sequence as a context sequence for another RS with the three simple reactions: $(\{0\}, \{1\}, \{0\})$, $(\{1\}, \{0\}, \{1\})$, $(\{-1\}, \{0\}, \{-1\})$.

The RS was designed so that the trigger can be applied at any step, and if the trigger is at a step that is not in a zero state (as is the case in step 10), the RS will remember that the trigger was there even though it is no longer there in the next context sequence step, and in this case, the RS is designed to still switch behavior.

When the process is oscillating between 0 and 1, there is no way for it to switch to oscillating between 0 and -1 without the interaction of the trigger from the context sequence. This behavior of two modes is depicted in Figure 3.2 by the presence of two subgraphs.

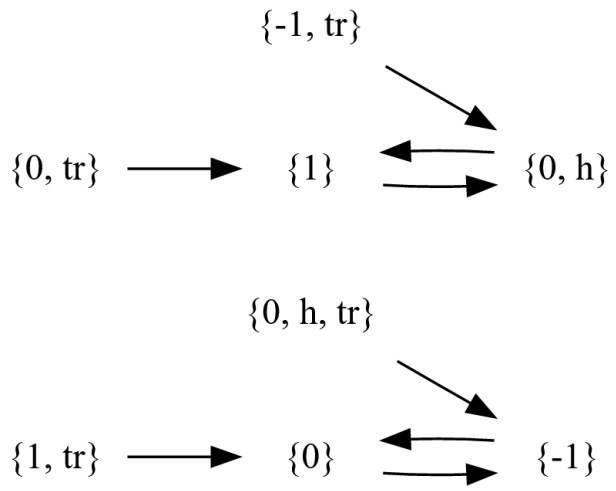

Figure 3.2: Directed graph for Example 3.1.1.

3.1.3 Interpretations of interactive processes

Some biochemical systems, such as the eukaryotic heat shock response of cells, can be modeled as an interactive process (see [1]). This reaction system (a model of the heat shock response taking place in an individual cell) is part of a bigger system (the entire organism and the external environment). The bigger system interacts with the RS defined in [1] by introducing or removing heat stress through the context sequence. This is an example of a common interpretation of interactive processes.

Another interpretation, mentioned in [8], is described as the result sequence representing our knowledge about the system that is being modeled, and the state sequence represents observations of the system. Each observation predicts the next.

Also described in [8], being able to associate a transition system (a finite automaton) with a reaction system is another implication of interactive processes. The transition system of a RS is defined below.

Definition 3.2. Let $\mathcal{A} = (S, A)$ be a RS. The *transition system of* $\mathcal{A}$, denoted by $tr(\mathcal{A})$, is the ordered pair (Q, τ) where the set of states $Q = \mathcal{P}(S)$, and the transition relationship $\tau \subseteq Q \times Q$ is defined by: for $X, Y \in Q, (X, Y) \in \tau$ if and only if $res_{\mathcal{A}}(X) \subseteq Y$ [8].

3.1.4 Sequences generated by reaction systems

In an interactive process π of RS $\mathcal{A} = (S, A)$ with interaction sequence C_i and result sequence D_i, the notation used in [8] to denote the transition from one step to the next is

$$\pi : C_0 \to (D_1, C_1) \to \cdots \to (D_n, C_n)$$

A process is described in [3] which has the initial state $W_0 = C_0$, but otherwise, if $C_i \subseteq D_i$, $1 \leq i \leq n$, then we have an equivalent process π to one where each $C_i = \varnothing$. If this is the case, then π is called context-independent .

In a context-independent process, we can describe the state transitions, starting with initial state W_0, with the notation:

$$W_0 \to_{\mathcal{A}} D_1 \to_{\mathcal{A}} D_2 \to_{\mathcal{A}} \ldots \to_{\mathcal{A}} D_m$$

The sequence $D_1, D_2, \ldots, D_m$ is called a sequence of length m generated by the reaction system $\mathcal{A}$. According to [20], for large enough m, one of the following always occurs at some point in the sequence D_i.

1. $D_m = \varnothing$ at which point $en_{\mathcal{A}}(D_m) = \varnothing$, and in this case the sequence is a *terminating sequence* of length m.
2. $D_m = D_j$ for some $j < m$, in which case the sequence contains a *cycle* of length m.

The sequence generated by the reaction system in Example 3.1.2 (as shown, it will produce a cycle, but by removing reaction 6, it produces a terminating sequence) is an example of a terminating sequence. Every reaction in the example RS is inhibited by the entity X, so when that entity shows up in $res_{\mathcal{A}}$, the next step in the sequence is $\varnothing$ and the process stops. The sequence terminates.

Some of the theorems in this section make a distinction between terminating sequences and cycles. However, when there is no need to make that distinction, we could say that a terminating sequence is a cycle of length one since $\varnothing \to \varnothing$.

There are several known facts about the lengths of cycles and terminating sequences

which reaction system are capable of generating. For a given background set S, the notion of maximally inhibited reactions, defined in [16], makes it possible to easily construct a RS which generates either a cycle or terminating sequence that includes every state in $\mathcal{P}(S)\backslash\{S, \varnothing\}$ in any order we may choose.

Definition 3.3. A reaction over the background set S is *maximally inhibited* if it is of the form $R, S\backslash R, P$. A reaction system is maximally inhibited if all of its reactions are maximally inhibited.

Theorem 3.4. *Given the base set S with n elements, there exists effectively a reaction system with a terminating state sequence of length $2^n - 1$, as well as a reaction system with a cycle of length $2^n - 2$. Moreover, the reaction systems can be constructed in such a way that the elements in the terminating state sequence and cycle are in any preassigned order.*

The proof in [16] for Theorem 3.4 is a simple construction proof. For a given background set S and the desired state transition sequence $T_1, T_2, ..., T_n$ where $n = |\mathcal{P}(S) - 2|$, then for each transition $T_i \to T_{i+1}, i = 1, 2, ..., n-1$ a maximally inhibited reaction a_i is constructed with $R_a = T_i, I_a = S\backslash R,$ and $P_a = T_{i+1}$. If we are constructing a RS for a cycle, then the last reaction, a_n, maps back to T_1. For a terminating sequence, there is no a_n needed.

Example 3.1.2. For the background set $S = \{a, b, c\}$ and state transition sequence $\{a\} \to \{a, b\} \to \{b\} \to \{a, c\} \to \{c\} \to \{b, c\} \to \{a\}$, the maximally inhibited reactions for the RS which generates this cycle are:

$$a_1 = (\{a\}, \{b, c\}, \{a, b\}) \qquad a_4 = (\{a, c\}, \{b\}, \{c\})$$
$$a_2 = (\{a, b\}, \{c\}, \{b\}) \qquad a_5 = (\{c\}, \{a, b\}, \{b, c\})$$
$$a_3 = (\{b\}, \{a, c\}, \{a, c\}) \qquad a_6 = (\{b, c\}, \{a\}, \{a\}).$$

By removing a_6 from the reaction set, we will get the terminating sequence instead. Figure 3.3 is a visual depiction of the cycle and terminating sequence just described.

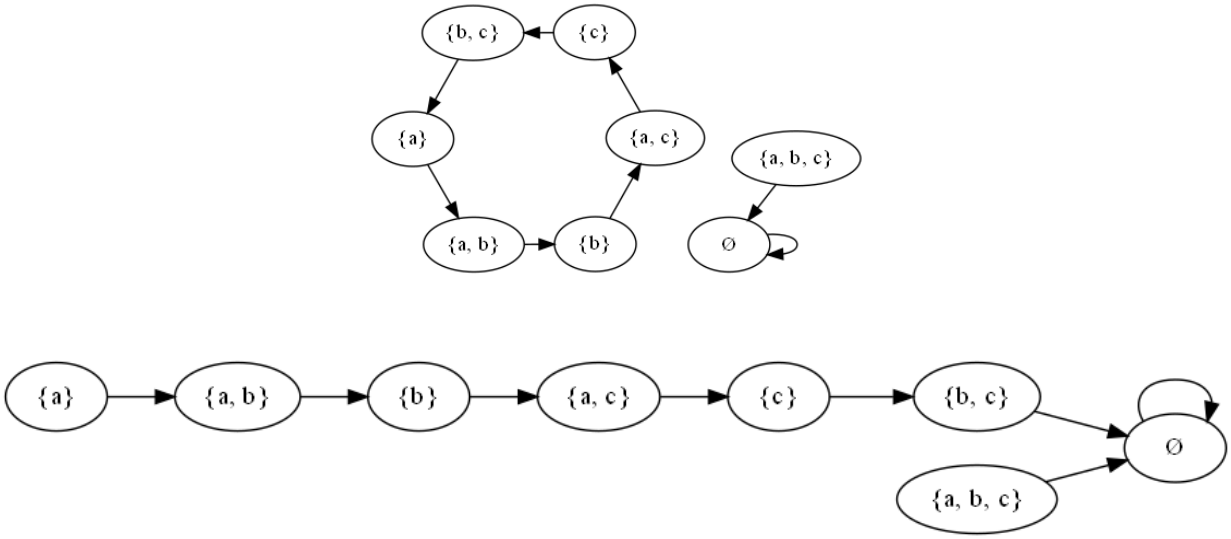

Figure 3.3: The cycle and terminating sequence for Example 3.1.2.

Although a large maximally inhibited RS may seem bloated, they can usually be simplified using the minimization techniques discussed in Chapter 5, and the functionally equivalent RS is much smaller. See Appendix A.2 for an example.

In [3], sequences generated by reaction systems are described as potentially having elaborate behavior and capability to generate long sequences. As seen in the following theorem summarized in [3] and proved in [7], the length of a sequence (whether terminating or cyclic) can be exponential based on certain (r, i) classes of the RS function. Although there are advantages to reaction systems of small classes such as $(1, 1)$, this theorem shows one of the limitations of smaller classes. The following theorem shows a much greater lower bound capability for the length of cycles and terminating sequences for higher classes of reaction system.

Theorem 3.5. *Let $k \geq 0$ be an integer. Then:*

1. *There exists a $(3k, 2)$ reaction system with a $(3 \cdot 2^k - 3)$-step terminating state sequence, and*
2. *There exists a $(3(k+1), 2)$ reaction system with a $(3 \cdot 2^k - 1)$-step cycle.*

The proof of Theorem 3.5 requires the following lemma proved in [7].

Lemma 3.6. *Let $n \geq 1$ be an integer, and let $\mathcal{A} = (S, A)$ be a reaction system with an n-step terminating state sequence. Then there exists a reaction system $\mathcal{A}' = (S', A')$ with a $(2n+3)$-step terminating state sequence of $\mathcal{A}'$.*

The proof of this lemma is by construction. Three new entities, $u, v,$ and w, are added to S to construct S', and five $(1,1)$ reactions are constructed and added to A to form A'. Hence, $A' \backslash A$ consists of only $(1,1)$ reactions which are constructed in such a way that $res^{2n+3}_{\mathcal{A}'}(\{u, v\} \cup T_0) = \varnothing$, where T_0 is the initial state of the n-step terminating sequence of $\mathcal{A}$. The proof details the iterations for this choice for the initial state and shows that $res_{\mathcal{A}'} \neq \varnothing$ until it reaches the $(2n+3)^{\text{th}}$ step.

Proof. Theorem 3.5, part 1. Let $k = 0$, then a RS of class $(3k, 2) = (0, 2)$ has no enabled reactions, and so $res_{\mathcal{A}}$ is always $\varnothing$. Hence, the RS has a $(3 \cdot 2^k - 3)$-step terminating sequence (i.e. a 0-step sequence). Thus, the statement is true for $k = 0$.

Assume that for every integer $k \geq 0$ there exists a $(3k, 2)$ reaction system $\mathcal{A}$ with a $(3 \cdot 2^k - 3)$-step terminating sequence. We need to show that there exists a $(3(k+1), 2)$ reaction system with a $(3 \cdot 2^{(k+1)} - 3)$-step terminating sequence. When applying Lemma 3.6 for $\mathcal{A}$ and $n = 3 \cdot 2^k - 3$, the lemma constructs $\mathcal{A}'$ with a $(2n+3)$-step terminating sequence. Thus, the terminating sequence of $\mathcal{A}'$ has $2n + 3 = 2(3 \cdot 2^k - 3) + 3 = 3 \cdot 2^{(k+1)} - 3$ steps. □

The proof of Theorem 3.5, part 2, is very similar. Given a RS with an n-step terminating sequence, a lemma similar to Lemma 3.6 shows how to construct a RS with an $(n+2)$-step cycle, and then it is shown by induction that there is a cycle of length $n = 3 \cdot 2^k - 3$.

The next theorem discussed in [3] and proved in [6], is of interest because it provides a way to design a reaction system by controlling the cardinalities $|S| = s, |R| = r, |I| = i$, $|P| = p,$ and a proportion μ so that the state sequence has a low probability of terminating.

Theorem 3.7. *Consider a background set of size s and a state T that contains a certain proportion, μ, of entities from the background set, i.e. $|T| = \mu s$. For positive integers*

r, i, and p (with $r + i \leq n$ and $p \leq n$), the probability that a random (r, i, p) reaction is enabled by T is given by the following closed formula using binomial coefficients:

$$\frac{\binom{\mu s}{r}\binom{(1-\mu)s}{i}}{\binom{s}{r}\binom{s-r}{i}} \tag{3.1}$$

Proof. The denominator of the formula is the number of all possible (r, i) reactions, i.e. r reactants are chosen from s entities, and from the remaining entities numbering $s - r$, i inhibitors are chosen. The numerator is the number of possible enabled reactions when the current state has μs entities, i.e. r reactants are chosen from μs entities, and from the remaining entities numbers $s - \mu s = (1 - \mu)s$, i inhibitors are chosen. □

The limit of Formula 3.1 as $s \to \infty$ is $\mu^r(1-\mu)^i$, and so this simpler formula can be used when the background set is large.

According to the limit formula, for a $(1, 1)$ RS over background set S with a large $|S|$, the probability that a random reaction will be enabled by T when $|T| = \mu|S|$ is given by $\mu^1(1-\mu)^1 = \mu - \mu^2$. This probability is maximized when $\mu = 0.5$, hence the highest probability that a $(1, 1)$ RS does not terminate is 0.25. Clearly, higher (r, i) classes have greater potential to generate non-terminating sequences, nevertheless this shows that the probability of a $(1, 1)$ RS not terminating is not negligible.

3.2 Functions defined by reaction systems

We mentioned the *res* function in Chapter 2. In [18], RS functions are described in more detail. A function defined by a reaction system $\mathcal{A} = (S, A)$ is a mapping $\mathcal{P}(S) \to \mathcal{P}(S)$. If f is such a mapping, then it is a representation of the mapping $res_{\mathcal{A}}$.

Definition 3.8. Consider a reaction system $\mathcal{A}$ over a finite set S. Denote by S_1 (resp. S_2) the set of all nonempty (resp. proper nonempty) subsets of S. (Thus, the cardinalities of S_1 and S_2 are $2^{|S|} - 1$ and $2^{|S|} - 2$, respectively.) Consider the function $f_{\mathcal{A}}$ defined as follows, for

$Y \subseteq S$. If $res_{\mathcal{A}}(Y) = Y' \neq \varnothing$, then $f_{\mathcal{A}}(Y) = Y'$. If $res_{\mathcal{A}}(Y) = \varnothing$, then $f_{\mathcal{A}}(Y)$ is undefined. The function $f_{\mathcal{A}}$ is termed total if it is defined for all elements of S_2.

As described in [5], [10], and Section 5.1.1, there are two "boundary conditions" which characterize RS functions. A function $f : \mathcal{P}(S) \to \mathcal{P}(S)$ is an RS function if and only if $f(\varnothing) = f(S) = \varnothing$. This statement is equivalent to Proposition 5.2 (see also [10]) which is part of a discussion about viewing a reaction system as a graph.

Example 3.2.1. This example shows two small reaction systems $\mathcal{A} = (S, A)$ and $\mathcal{B} = (S, B)$ over a background set $S = \{a, b, c\}$ and the set functions defined by them. $\mathcal{A}$ has the property that all reactions have exactly one reactant, and $\mathcal{B}$ has the property that all reactions have exactly one inhibitor. These properties will be further discussed in Section 4.1.

Table 3.4: Examples of reaction systems and functions defined by them.

$$\begin{array}{l} a_1 = (\{a\}, \{b, c\}, \{b\}) \\ a_2 = (\{b\}, \{a, c\}, \{a\}) \\ a_3 = (\{c\}, \{a\}, \{a\}) \\ a_4 = (\{c\}, \{b\}, \{b\}) \end{array} \quad f_{\mathcal{A}}(T) = \begin{cases} \{a\} & , T \in \{\{b\}, \{b, c\}\} \\ \{b\} & , T \in \{\{a\}, \{a, c\}\} \\ \{a, b\} & , T = \{c\} \\ \varnothing & , \text{otherwise.} \end{cases}$$

$$\begin{array}{l} b_1 = (\{a, b\}, \{c\}, \{a\}) \\ b_2 = (\{a, c\}, \{b\}, \{b\}) \\ b_3 = (\{b\}, \{a\}, \{c\}) \\ b_4 = (\{b\}, \{c\}, \{b\}) \end{array} \quad f_{\mathcal{B}}(T) = \begin{cases} \{a, b\} & , T = \{a, b\} \\ \{b\} & , T = \{a, c\} \\ \{c\} & , T = \{b, c\} \\ \{b, c\} & , T = \{b\} \\ \varnothing & , \text{otherwise.} \end{cases}$$

3.3 Boolean functions

There is a close relationship between RS functions and Boolean functions. Translating an RS function into a Boolean vector function is instrumental in simplifying/minimizing the RS using logic minimization algorithms.

A reaction system $\mathcal{A} = (S, A)$, where $S = \{a, b, c, d\}$ and $A = \{a_1, ..., a_{10}\}$ is shown in Table 3.5. This RS can be represented as a Boolean vector function. Sometimes, instead,

an RS will be discussed as being equivalent to a set of Boolean functions.

More generally, for a RS $\mathcal{A} = (S, A)$, where $S = \{s_1, ..., s_n\}$, $A = \{a_1, ..., a_m\}$, and $T \subseteq S$, let $F : [0,1]^n \to [0,1]^n$ be a function with Boolean variable $x_1, ..., x_n$, where $n = |S|$ and the x's are defined as:

$$x_i = \begin{cases} 1, & s_i \in T \\ 0, & s_i \notin T \end{cases}, i = 1, 2, ..., n.$$

Consider $res_{\mathcal{A}}(T)$. Let X and Y be Boolean vectors. Let $X = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}$ represent the input T to RS $\mathcal{A}$, then $Y = F(X)$ is the Boolean vector representing $res_{\mathcal{A}}(T)$. The mapping $X \to Y$ is defined by a set of logic functions $f_1, ..., f_n$. Thus, $y_i = f_i(X)$.

All possible inputs for $\mathcal{A}$, i.e. $\mathcal{P}(S)$, have corresponding values for X. Thus, a truth table for F can be generated with each row having the form:

x_1	x_2	...	x_n	$f_1(X)$	$f_2(X)$	...	$f_n(X)$

The following shows, by example, how a reaction system can be converted to an equivalent set of Boolean functions.

Example 3.3.1. Table 3.5 shows an example of a RS $\mathcal{A} = (S, A)$ where $S = \{a, b, c, d\}$. For each entity $s_i \in S$, we associated s_i with the variable x_i of input vector X, and we associate s_i with the Boolean function f_i. Then each f_i contains one term for each reaction which is capable of producing s_i as a product. Within each of those terms, there is one literal x_i for each reactant s_i in the reactant set, and one negated literal $\overline{x_i}$ for each inhibitor s_i in the inhibitor set. In this example, in the reaction set A, no product sets contain a, hence $f_1(X) = \varnothing$ in Table 3.6. But for entity b, we see that there are six reactions such that $b \in P_a$, hence we see six terms in f_2 with the literals as described above. This process translates RS $\mathcal{A}$ into its equivalent disjunctive normal form Boolean vector function $F(x)$ which is shown in Table 3.6.

The truth table for the RS in Table 3.5 is shown in Table 3.7 with two extra columns: the first column shows the set of entities represented by the x_i's (i.e. $T \subseteq \mathcal{P}(S)$), and the last column shows the set represented by the f_i's (i.e. $res_{\mathcal{A}}$).

Table 3.5: Reactions of A for RS $\mathcal{A}$.

$$
\begin{array}{ll}
a_1 = (\{b,c,d\},\{a\},\{c,d\}) & a_6 = (\{a,c\},\{b,d\},\{c\}) \\
a_2 = (\{b,c\},\{a,d\},\{c,d\}) & a_7 = (\{b\},\{a,c,d\},\{b,c,d\}) \\
a_3 = (\{b,d\},\{a,c\},\{b,c,d\}) & a_8 = (\{a,c,d\},\{b\},\{c\}) \\
a_4 = (\{a,d\},\{b,c\},\{b,c\}) & a_9 = (\{a,b\},\{c,d\},\{b\}) \\
a_5 = (\{a\},\{b,c,d\},\{b,c\}) & a_{10} = (\{a,b,d\},\{c\},\{b\})
\end{array}
$$

Table 3.6: Boolean vector function for RS $\mathcal{A}$ shown in Table 3.5.

$$
\begin{aligned}
F(X) = \quad & \begin{bmatrix} f_1(X) & f_2(X) & f_3(X) & f_4(X) \end{bmatrix} \\
\text{where:} & \\
f_1(X) = \quad & \varnothing \\
f_2(X) = \quad & (x_2 \wedge x_4 \wedge \overline{x_1} \wedge \overline{x_3}) \vee (x_1 \wedge x_4 \wedge \overline{x_2} \wedge \overline{x_3}) \vee (x_1 \wedge \overline{x_2} \wedge \overline{x_3} \wedge \overline{x_4}) \vee (x_2 \wedge \overline{x_1} \wedge \\
& \overline{x_3} \wedge \overline{x_4}) \vee (x_1 \wedge x_2 \wedge \overline{x_3} \wedge \overline{x_4}) \vee (x_1 \wedge x_2 \wedge x_4 \wedge \overline{x_3}) \\
f_3(X) = \quad & (x_2 \wedge x_3 \wedge x_4 \wedge \overline{x_1}) \vee (x_2 \wedge x_3 \wedge \overline{x_1} \wedge \overline{x_4}) \vee (x_2 \wedge x_4 \wedge \overline{x_1} \wedge \overline{x_3}) \vee (x_1 \wedge x_4 \wedge \overline{x_2} \wedge \overline{x_3}) \vee \\
& (x_1 \wedge \overline{x_2} \wedge \overline{x_3} \wedge \overline{x_4}) \vee (x_1 \wedge x_3 \wedge \overline{x_2} \wedge \overline{x_4}) \vee (x_2 \wedge \overline{x_1} \wedge \overline{x_3} \wedge \overline{x_4}) \vee (x_1 \wedge x_3 \wedge x_4 \wedge \overline{x_2}) \\
f_4(X) = \quad & (x_2 \wedge x_3 \wedge x_4 \wedge \overline{x_1}) \vee (x_2 \wedge x_3 \wedge \overline{x_1} \wedge \overline{x_4}) \vee (x_2 \wedge x_4 \wedge \overline{x_1} \wedge \overline{x_3}) \vee (x_2 \wedge \overline{x_1} \wedge \overline{x_3} \wedge \overline{x_4})
\end{aligned}
$$

Table 3.7: Truth table.

$T \subseteq \mathcal{P}(S)$	x_1	x_2	x_3	x_4	f_1	f_2	f_3	f_4	$res_{\mathcal{A}}(T)$
$\varnothing$	0	0	0	0	0	0	0	0	$\varnothing$
$\{a\}$	1	0	0	0	0	1	1	0	$\{b, c\}$
$\{b\}$	0	1	0	0	0	1	1	1	$\{b, c, d\}$
$\{a, b\}$	1	1	0	0	0	1	0	0	$\{b\}$
$\{c\}$	0	0	1	0	0	0	0	0	$\varnothing$
$\{a, c\}$	1	0	1	0	0	0	1	0	$\{c\}$
$\{b, c\}$	0	1	1	0	0	0	1	1	$\{c, d\}$
$\{a, b, c\}$	1	1	1	0	0	0	0	0	$\varnothing$
$\{d\}$	0	0	0	1	0	0	0	0	$\varnothing$
$\{a, d\}$	1	0	0	1	0	1	1	0	$\{b, c\}$
$\{b, d\}$	0	1	0	1	0	1	1	1	$\{b, c, d\}$
$\{a, b, d\}$	1	1	0	1	0	1	0	0	$\{b\}$
$\{c, d\}$	0	0	1	1	0	0	0	0	$\varnothing$
$\{a, c, d\}$	1	0	1	1	0	0	1	0	$\{c\}$
$\{b, c, d\}$	0	1	1	1	0	0	1	1	$\{c, d\}$
$\{a, b, c, d\} = S$	1	1	1	1	0	0	0	0	$\varnothing$

Since reaction systems have a correspondence to Boolean functions, they have a natural correspondence to digital logic circuits. Since there has been tremendous interest in minimizing Boolean functions for the purpose of simplifying logic circuit designs, many Boolean function minimization methods have been developed. A discussion of many of the known methods for Boolean function minimization is in [13], and among the algorithms discussed is called Espresso. The Espresso algorithm stands out as being so successful that it has been included as the logic minimization tool used in nearly all contemporary logic synthesis tools. How we can use Espresso to minimize reaction systems is discussed in more detail in Chapter 5.

As an aside, since a reaction system can so naturally be represented as a set of logical formulas, this means it can also be represented as a logic circuit. As an example, consider the reaction system described in Table 3.5. The software RS Tools was used to convert the

reactions in Table 3.6 to Boolean functions (see Chapter 6 for a description of RS Tools and how it converts reactions into Boolean functions). The Boolean functions generated are shown in Table 3.6. These formulas are represented as the logic circuit diagram in Figure 3.4. This diagram was generated by entering the Boolean functions into software which optimizes the formulas and automatically generates an equivalent logic circuit diagram. The software used to generate Figure 3.4 is called Logic Friday.

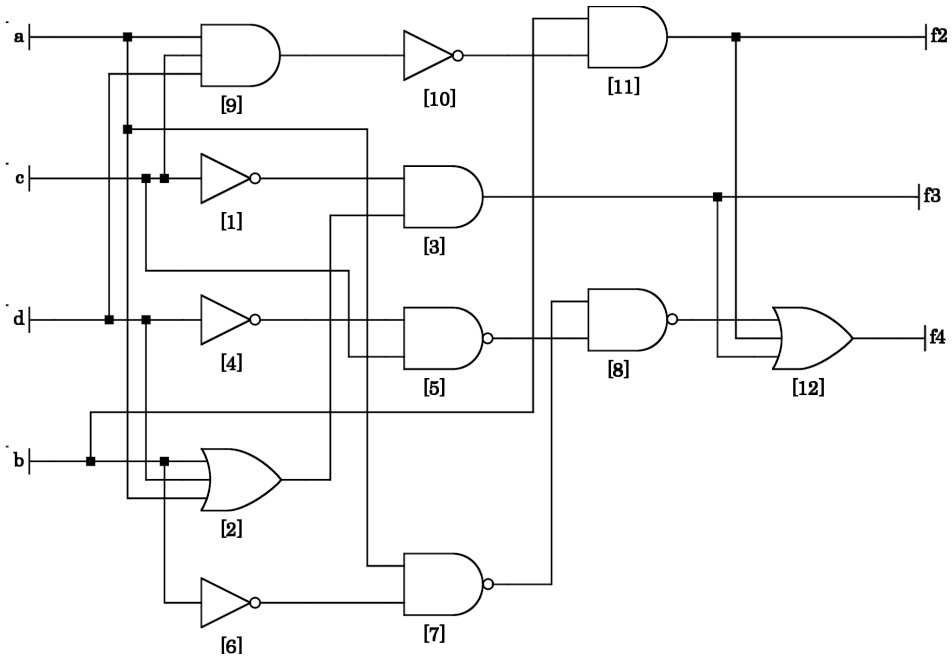

Figure 3.4: Logic circuit diagram.

4. Minimality of reaction systems

There are many facets to the minimality of reaction systems. Classifications and properties of reaction systems, especially where minimality is concerned, are discussed in this chapter.

4.1 Minimality and classes of reaction systems

As discussed in Section 2.4, classes of reaction systems are based on the cardinalities of the reactant and inhibitor sets, R_a and I_a, or the cardinality of their union which is notation M_a. In this section, we discuss reaction system classified as reactant-minimal, inhibitor-minimal, resource-minimal, product-minimal, and almost minimal as defined in [5], [18], and [19]. In [5], "minimal" reaction systems are defined as $(1, 1)$. To avoid confusion, this paper will refer to $(1, 1)$ reaction systems as resource-minimal so that we can make a clear distinction between $(1, 1)$ reaction systems and those that are minimized but not necessarily all the way to $(1, 1)$.

In addition, "almost minimal" reaction systems are defined. These classifications are based on the maximum cardinality of the sets of reactants, inhibitors, or both, i.e. resources.

Definition 4.1. Let $\mathcal{A} = (S, A)$ be a RS.

1. $\mathcal{A}$ is reactant-minimal if $|R_a| = 1$ for each $a \in A$.
2. $\mathcal{A}$ is inhibitor-minimal if $|I_a| = 1$ for each $a \in A$.
3. $\mathcal{A}$ is resource-minimal if $|M_a| = 2$ for each $a \in A$.

The terms in Definition 4.1 also apply to RS functions. So an RS function f is reactant-minimal (inhibitor-minimal, resource-minimal) if there is a reactant-minimal (inhibitor-minimal, resource-minimal) RS implementing f.

A product-minimal RS is defined in [19] to be a $(1, 1)$ RS where each of the reactions in the RS has a product set that is a singleton.

4.2 Subadditive properties of RS functions

Following is a description of set functions that have the union-subadditive or intersection-subadditive properties as defined in [5] and Definition 4.2 below. When RS functions have these properties, they can be used to provide a way to determine if reactions systems are of one of the minimality classes in Definition 4.1.

Definition 4.2. For a reaction system function $f : \mathcal{P}(S) \to \mathcal{P}(S)$ over a background set S:

1. f is *union-subadditive* if for any two subsets X and Y of S, $f(X \cup Y) \subseteq f(X) \cup f(Y)$, and
2. f is *intersection-subadditive* if for any two subsets X and Y of S, $f(X \cap Y) \subseteq f(X) \cup f(Y)$.

Example 4.2.1. Regarding the set functions defined below over $S = \{a, b, c, d\}$ and $T \subseteq S$, f_1 is union-subadditive, f_2 is intersection-subadditive, f_3 is neither, and f_4 is both.

$$f_1(T) = \begin{cases} \{a\}, & T \in \{\{b\}, \{b, c\}\} \\ \{b\}, & T \in \{\{a\}, \{a, c\}\} \\ \{a, b\}, & T = \{c\} \\ \varnothing, & \text{otherwise.} \end{cases} \qquad f_2(T) = \begin{cases} \{a, b\}, & T = \{a, b\} \\ \{b\}, & T = \{a, c\} \\ \{b, c\}, & T = \{b\} \\ \varnothing, & \text{otherwise.} \end{cases}$$

$$f_3(T) = \begin{cases} \{a\}, & T = \{a\} \\ \{b\}, & T = \{b\} \\ \{c\}, & T = \{a, b\} \\ \{d\}, & T = \{b, c\} \\ \varnothing, & \text{otherwise.} \end{cases} \qquad f_4(T) = \begin{cases} \{b, c\}, & T = \{a\} \\ \{c, d\}, & T = \{b\} \\ \{b, c, d\}, & T = \{a, b\} \\ \varnothing, & \text{otherwise.} \end{cases}$$

The definitions above are used to prove the following lemma which is instrumental in the characterization proofs in this chapter.

Lemma 4.3. *Let S be a finite set and f be a function $f : \mathcal{P}(S) \to \mathcal{P}(S)$.*

1. *f is union-subadditive if and only if for every non-empty $\mathcal{F} \subseteq \mathcal{P}(S)$, $f(\bigcup \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$.*
2. *f is intersection-subadditive if and only if for every for every non-empty $\mathcal{F} \subseteq \mathcal{P}(S)$, $f(\bigcap \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$.*

Proof. Let $\mathcal{F} \subseteq \mathcal{P}(S), \mathcal{F} \neq \varnothing$. Since S is finite, $\mathcal{F} = \{U_1, U_2, ..., U_k\}$ where $U_i \in \mathcal{P}(S), k = 1, 2, ..., 2^{|S|+1}$.

1. f is union-subadditive if and only if

$$\begin{aligned} f(\bigcup \mathcal{F}) &= f(U_1 \cup (U_2 \cup ... \cup U_k)) \\ &\subseteq f(U_1) \cup f(U_2 \cup U_3 \cup ... \cup U_k)) \\ &\subseteq f(U_1) \cup f(U_2) \cup f(U_3 \cup ... \cup U_k)) \\ &\subseteq ... \\ &\subseteq f(U_1) \cup f(U_2) \cup ... \cup f(U_k) \\ &= \bigcup f(\mathcal{F}). \end{aligned}$$

2. f is intersection-subadditive if and only if

$$\begin{aligned} f(\bigcap \mathcal{F}) &= f(U_1 \cap (U_2 \cap ... \cap U_k)) \\ &\subseteq f(U_1) \cup f(U_2 \cap ... \cap U_k)) \\ &\subseteq f(U_1) \cup f(U_2) \cup f(U_3 \cap ... \cap U_k) \\ &\subseteq ... \\ &\subseteq f(U_1) \cup f(U_2) \cup f(U_3 \cup ... \cup U_k)) \\ &= \bigcup f(\mathcal{F}). \end{aligned}$$

□

4.3 Characterization theorems

Given a reaction system $\mathcal{A}$ that is not apparent to be reactant-, inhibitor-, or resource-minimal, the theorems in this section, introduced and proved in [5], determine whether there exists a functionally equivalent reaction system, based on the subadditive properties of $f_{\mathcal{A}}$, that is in one of those special classes.

4.3.1 Reactant-minimal

Consider the reaction system, $\mathcal{C} = (C, S)$ over background set $S = \{a, b, c\}$ and with the reactions of C shown below with emphasis on reactant sets with more than one entity.

$$c_1 = (\mathbf{\{b,\ c\}}, \{a\}, \{a\}) \quad c_2 = (\{b\}, \{a, c\}, \{a\}) \qquad c_3 = (\mathbf{\{a,\ c\}}, \{b\}, \{b\})$$
$$c_4 = (\{a\}, \{b\}, \{b\}) \qquad c_5 = (\{c\}, \{a, b\}, \{a, b\})$$

At first glance, $\mathcal{C}$ does not appear to be reactant-minimal. Since $\mathcal{P}(S)$ is so small, it is easy to check if $f_{\mathcal{C}}$ is union-subadditive. This is done in Table 4.1. Some pairs of subsets are not in the table since they do not need to be checked. For example, $X = \{a\}, Y = \{c\}$ is checked so we do not need to check the pair $X = \{c\}, Y = \{a\}$. We see that $f_{\mathcal{C}}$ is indeed union-subadditive. Alternatively,we could have used software, such as RS Tools, to verify union-subadditivity.

Table 4.1: Verification of union-subadditivty of $f_{\mathcal{C}}$

X	Y	$f_{\mathcal{C}}(X \cup Y)$	$f_{\mathcal{C}}(X) \cup f_{\mathcal{C}}(Y)$	Subadditive?
$\{a\}$	$\{a\}$	$\{b\}$	$\{b\}$	✓
$\{a\}$	$\{c\}$	$\{b\}$	$\{a, b\}$	✓
$\{b\}$	$\{c\}$	$\{a\}$	$\{a, b\}$	✓

Theorem 4.5 below allows us to conclude that there exists a reactant-minimal reaction system that is functionally equivalent to $\mathcal{C}$. In fact, $\mathcal{A}$ in Example 3.2.1, which is clearly reactant-minimal, is functionally equivalent to $\mathcal{C}$.

Lemma 4.4. *For a finite set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be a union-subadditive function. Let $T \subseteq S$, $T \neq \varnothing$, and let $q \in f(T)$. If there exists an $x \in T$ such that for every $U \subseteq T$, $x \in U$, then $q \in f(U)$.*

Proof. Assume, to the contrary, that f is union-subadditive, $T \subseteq S, T \neq \varnothing$, and that $q \in f(T)$ such that for every $x \in T$, there exists a $U \subseteq T$ such that $x \in U$ but $q \notin f(U)$.

For each $x \in T$ we choose any not necessarily unique U_x which meets the conditions above. Let $\mathcal{F} = \{U_x : x \in T\}$. We observe the following.

1. Since $x \in T$ then $x \in U_x \subseteq T$, so it follows that $\bigcup \mathcal{F} = T$.
2. By our assumption, $q \in f(T)$.
3. Also by our assumption, for every $x \in T$,

$$q \notin f(U_x) \implies q \notin \bigcup f(U_x) \implies q \notin f(\bigcup \mathcal{F}).$$

Thus, by observations 1 and 2, $q \in f(\bigcup \mathcal{F})$, and by observation 3, $q \notin f(\bigcup \mathcal{F})$, which is a contradiction. Therefore, the original claim holds. □

Theorem 4.5. *For a finite non-empty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be a reaction system function. Then f is reactant-minimal if and only if f is union-subadditive.*

Proof. First we will show that f is reactant-minimal implies f is union-subadditive. Let $\mathcal{A} = (S, A)$ be a reactant-minimal RS and let f be the RS function defined by $\mathcal{A}$. Let $T, U \subseteq S$. We will address the two cases: $res_{\mathcal{A}}(T \cup U) \neq \varnothing$ and $res_{\mathcal{A}}(T \cup U) = \varnothing$.

For the first case, assume $res_{\mathcal{A}}(T \cup U) \neq \varnothing$, then there exists an $a \in A$ such that $en_a(T \cup U)$. Let $q \in res_a(T \cup U)$, and let $x \in R_a$, i.e. $R_a = \{x\}$, a single reactant since $\mathcal{A}$ is reactant-minimal. If $x \in T$, then $en_a(T)$ and hence $q \in res_{\mathcal{A}}(T)$. If $x \in U$, then $en_a(U)$ and hence $q \in res_{\mathcal{A}}(U)$. Thus,

$$q \in res_{\mathcal{A}}(T) \cup res_{\mathcal{A}}(U) = f(T) \cup f(U).$$

The other case is that if $res_{\mathcal{A}}(T \cup U) = \varnothing$, then

$$res_{\mathcal{A}}(T \cup U) = f(T \cup U) = \varnothing \subseteq f(T) \cup f(U).$$

In both cases, $f(T \cup U) \subseteq f(T) \cup f(U)$ and therefore f is union-subadditive.

Now we will show that f is union-subadditive implies f is reactant-minimal. Let f be union-subadditive. By applying Lemma 4.4 we construct a set of reactions A and a reactant-minimal RS in the following manner.

1. For each $T \subset S$, $T \neq \varnothing$, and for every $q \in f(T)$:
   (a) We choose one $x \in T$, hereafter referred to as x_T^q, such that for every $U \subseteq T$, if $x_T^q \in U$, then $q \in f(U)$.
   (b) Define the reaction $a_T^q = (\{x_T^q\}, S \backslash T, \{q\})$.

2. Let $A = \{a_T^q : T \subseteq S \text{ and } q \in f(T)\}$.
3. Define $\mathcal{A} = (S, A)$. Since each a_T^q has one reactant, by definition $\mathcal{A}$ is a reactant-minimal RS.

Note that we excluded $T \in \{\varnothing, S\}$ above since reactions are not enabled for these subsets.

For the RS $\mathcal{A}$ just constructed, we show that $res_{\mathcal{A}} = f$ by showing that for every $T \subseteq S$, $f(T) \subseteq res_{\mathcal{A}}(T)$ and $res_{\mathcal{A}}(T) \subseteq f(T)$.

1. Let $T \subseteq S$ and $q \in f(T)$, then $a_T^q = (\{x_T^q\}, S\backslash T, \{q\}) \in A$ and $x_T^q \in T$ due to the way A was constructed. Hence, a_T^q is enabled by T and so $q \in res_{\mathcal{A}}(T)$. Thus, since this holds for every $q \in f(T)$,
$$f(T) \subseteq res_{\mathcal{A}}(T).$$
2. Let $T \subseteq S$ and $q \in res_{\mathcal{A}}(T)$. We do not know that $q \in res$ of a_T^q necessarily, but since $q \in res_{\mathcal{A}}(T)$, there exists an $a \in A$ for some $W \subseteq S$ such that $a = a_W^q$ and $en_a(T)$. From this we know that $T \subseteq W$ (if not, there exists a $y \in T \cap (S\backslash W)$, and since $I_a \in (S\backslash W)$, then $y \in T \cap I_a$ in which case a would not be enabled which is a contradiction). We know that $x_W^q \in T$ since $\{x_W^q\} = R_a$ and so $en_a(T)$, and also $P_a = \{q\}$, consequently $q \in f(T)$. Thus, since this holds for every $q \in res_{\mathcal{A}}(T)$,
$$res_{\mathcal{A}}(T) \subseteq f(T).$$

We have shown that for every $T \subseteq S$, $f(T) \subseteq res_{\mathcal{A}}(T)$ and $res_{\mathcal{A}}(T) \subseteq f(T)$, therefore $res_{\mathcal{A}} = f$, and since $\mathcal{A}$ is reactant-minimal, so is f.

We have shown that f is reactant-minimal implies f is union-subadditive and f is union-subadditive implies f is reactant-minimal, therefore the proof is complete. □

4.3.2 Inhibitor-minimal

The reaction system $\mathcal{D} = (D, S)$ over background set $S = \{a, b, c\}$ has the following reactions.

$$d_1 = (\{a, b\}, \{c\}, \{a, b\}) \quad d_2 = (\{a, c\}, \{b\}, \{b\})$$
$$d_3 = (\{b, c\}, \{a\}, \{c\}) \quad d_4 = (\{b\}, \mathbf{\{a,\ c\}}, \{c, b\})$$

Reaction system $\mathcal{D}$ appears to be neither reactant-minimal nor inhibitor-minimal. Once again, since $\mathcal{P}(S)$ is so small, we could verify subadditive properties of RS function $f_{\mathcal{D}}$ but henceforth we will rely on RS Tools to check such properties. In fact, RS $\mathcal{D}$ is intersection-subadditive (incidentally, it is not union-subadditive). The next theorem tells us there

exists an inhibitor-minimal RS that is functionally equivalent to RS $\mathcal{D}$. In fact, RS $\mathcal{D}$ is functionally equivalent to RS $\mathcal{B}$ in Example 3.2.1 which is immediately apparent as being inhibitor-minimal.

Theorem 4.6. *For a finite non-empty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be an RS function. Then f is inhibitor-minimal if and only if f is intersection-subadditive.*

The following definition and lemma will be used in the proof of Theorem 4.6.

Definition 4.7. Let f be an RS function over S, $V \subseteq S$, $y \in S \backslash V$, and $q \in f(V)$. Then y is *f-excluding for V,q* if for every $W \subseteq S$, $V \subseteq W$ and $y \notin W$ implies $q \in f(W)$. We use *excluding(f, V, q)* to denote the set of f-excluding entities for V, q.

Lemma 4.8. *Let $V \subseteq S$ and $f(V)$ be an intersection-subadditive RS function over S. Then for every $q \in S \backslash V$, there exists a $y \in S \backslash V$ such that $y \in excluding(f, V, q)$.*

Proof. Let $T \subseteq S$ and f be an intersection-subadditive RS function over S, and let $q \in f(T)$. Assume, contrary to the claim, that for every $y \notin T$, there exists a $W \subseteq S$ such that $T \subseteq W$ and $y \notin W$ and $y \notin excluding(f, T, q)$. Hence $q \notin f(W)$ by Definition 4.7. For each $y \notin T$, we choose a not necessarily unique set W, denoted by W_y, that meets the condition above for W. Thus,

$$y \notin T \implies y \notin W_y \implies \textstyle\bigcap_{y \notin T} W_y = T.$$

Since $q \in f(T)$, then

$$q \in f(\textstyle\bigcap_{y \notin T} W_y).$$

Also,

$$q \notin f(W_y) \implies q \notin \textstyle\bigcup_{y \notin T} f(W_y).$$

Consequently,

$$f(\textstyle\bigcap_{y \notin T} W_y) \nsubseteq \bigcup_{y \notin T} f(W_y)$$

which is a contradiction since we assumed that f is intersection-subadditive. Therefore the claim holds. □

Proof (Theorem 4.6). First we will show that f is inhibitor-minimal implies f is intersection-subadditive. Let $\mathcal{A} = (S, A)$ be an inhibitor-minimal RS, and let f be a RS function defined by $\mathcal{A}$. Let $\mathcal{F} \subseteq \mathcal{P}(S)$. If $\mathcal{A}$ is not enabled on $\bigcap \mathcal{F}$ then,

$$en_{\mathcal{A}}(\textstyle\bigcap \mathcal{F}) = \varnothing \implies f(\bigcap \mathcal{F}) = \varnothing \subseteq \bigcup f(\mathcal{F})$$

thus f is intersection-subadditive.

So then we assume $\mathcal{A}$ is enabled on $\bigcap \mathcal{F}$, hence $f(\bigcap \mathcal{F}) \neq \varnothing$ and there exists a reaction $a = (R_a, I_a, P_a) \in A$ such that $en_a(\bigcap \mathcal{F})$. Since $\mathcal{A}$ is inhibitor-minimal, I_a is a singleton, i.e. $I_a = \{y\}$ such that $y \in S \backslash R_a$. Let $q \in P_a$, then

$$P_a \subseteq f(\bigcap \mathcal{F}) \implies q \in f(\bigcap \mathcal{F}) \implies q \in res_{\mathcal{A}}(\bigcap F). \tag{4.1}$$

Let $T \in \mathcal{F}$ such that $y \notin T$ (we know such a T exists since $en_a(\bigcap F)$ implies $I_a = y \notin \bigcap \mathcal{F}$). We also know $en_a(\bigcap \mathcal{F})$. Thus,

$$R_a \subseteq \bigcap \mathcal{F} \subseteq T.$$

Since $R_a \subseteq T$, then

$$en_a(\bigcap F) \implies en_a(T) \implies q \in res_a(T) \implies q \in res_{\mathcal{A}}(T). \tag{4.2}$$

Consider now $\mathcal{F}$, specifically $\bigcup_{V \in \mathcal{F}} res_{\mathcal{A}}(V)$. From the fact that one of the V's is T and from (4.2):

$$q \in \bigcup_{V \in \mathcal{F}} res_{\mathcal{A}}(V) = \bigcup res_{\mathcal{A}}(\mathcal{F}) = \bigcup f(\mathcal{F}).$$

So from (4.1) we have $q \in f(\cap \mathcal{F})$ and from (4.2) $q \in \bigcup f(\mathcal{F})$, therefore, by definition, f is intersection-subadditive.

We will now show that f is intersection-subadditive implies f is inhibitor-minimal. Let f be intersection-subadditive and let $\mathcal{A} = (S, A)$ such that

$$A = \{(T, \{y\}, \{q\}) : T \subseteq S, q \in f(T), \text{ and } y \in excluding(f, T, q)\}. \tag{4.3}$$

Hence, $\mathcal{A}$ is constructed to be inhibitor-minimal. We will show that f is an RS function defined by $\mathcal{A}$.

We will first show that for every $T \subseteq S, f(T) \subseteq res_{\mathcal{A}(T)}$. Let $T \subseteq S$ and $q \in f(T)$. By Lemma 4.8 and the construction of $\mathcal{A}$ we know there exists a reaction $a = (T, \{y\}, \{q\}) \in A$ such that $y \in excluding(f, T, q)$. Since y is a member of the f-excluding entities for T, q, reaction a will not be inhibited, hence

$$en_a(T) \text{ and } P_a = \{q\} \implies q \in res_a(T) \implies q \in res_{\mathcal{A}}(T). \tag{4.4}$$

Thus, from the assumption that $q \in f(T)$ leading to (4.4) , we have

$$q \in f(T) \text{ and } q \in res_{\mathcal{A}}(T) \implies f(T) \subseteq res_{\mathcal{A}}(T). \tag{4.5}$$

Next, we will show that for every $T \subseteq S, res_{\mathcal{A}}(T) \subseteq f(T)$. Let $U \subseteq S$ and $q \in res_{\mathcal{A}}(U)$. Then there exists a reaction $a \in A$ such that $en_a(U)$ and $P_a = \{q\}$. We know from (4.3) and Definition 4.7 that for some T such that $U \subseteq T \subseteq S$ and $y \notin T$, $a = (T, \{y\}, \{q\})$ and

$y \in excluding(f, T, q)$. Thus,

$$q \in f(T) \implies res_{\mathcal{A}}(T) \subseteq f(T). \tag{4.6}$$

We have shown that for every $T \subseteq S$, (4.3) and (4.6) hold, thus $f(T) = res_{\mathcal{A}}(T)$, i.e. f is defined by $\mathcal{A}$ which was constructed to be inhibitor-minimal, hence f is inhibitor-minimal.

This completes the proof that f is inhibitor-minimal if and only if f is intersection-subadditive. □

4.3.3 Resource-minimal

The function defined by a RS with the reactions shown in Table 3.5 has both union- and intersection-subadditive properties. As we will see, because it has both subadditive properties, it has a functionally equivalent resource-minimal RS that is much simpler.

Before we get to the characterization theorem for arbitrary resource-minimal reaction system, we first need to discuss several prerequisite theorems. In particular, we first need to provide a characterization theorem for RS focus functions.

Resource-minimal focus function defined by an RS

Definition 4.9. A function $f : \mathcal{P}(S) \to \mathcal{P}(S)$ for a finite set S is a *focus function* if there is an element $q \in S$, referred to as the *focus of* f, such that the range of f is $\{\varnothing, q\}$. The focus of f is unique.

Let $T \subseteq S$. T is *focused by* f if $f(T) \neq \varnothing$ (i.e., $f(T) = q$). The notation $focused(T)$ refers to the set $\{T \subseteq S : T \text{ is focused by} f\}$.

Let $x \in T$ and $U \subseteq T$. If $f(U) \neq \varnothing$ when $x \in U$ for every $U \subseteq T$, then x is *f-special for* T. The notation $special(f, T)$ refers to the set $\{x \in T : x \text{ is } f\text{-special}\}$.

Let $y \in T, y \neq x$. If $f(U) \neq \varnothing$ for every $U \subseteq S$ where $x \in U$ and $y \notin U$, then y is an *f-partner of* x. The notation $partner(f, x)$ refers to the set $\{y \in S : y \text{ is an } f\text{-partner of} x\}$.

The following lemmas, originally proved in [5], are needed for the proof of the main theorem in this section.

Lemma 4.10. *For a finite non-empty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be a focus function that is both union- and intersection-subadditive. For every non-empty $T \subseteq S$, if $T \in focused(f)$ then there exists an $x \in T$ such that $x \in special(f, T)$.*

Proof. Assume that there exists a $T \subseteq S$ such that $T \in focused(f)$ and, contrary to the claim, there does not exists an $x \in T$ such that $x \in special(f, T)$. That is, for every $x \in T$,

there exists $U_x \subseteq T$ (U_x is not necessarily unique) such that $x \in U_x$ and $f(U_x) = \varnothing$. Since for every $x \in T, x \in U_x$ and $U_x \subseteq T$, it follows that $\bigcup_{x \in T} U_x = T$. Since T is focused by f, then

$$f(\bigcup_{x \in T} U_x) = f(T) = \{q\}. \tag{4.7}$$

By our assumption, for every $x \in T, f(U_x) = \varnothing$, hence

$$\bigcup_{x \in T} f(U_x) = \varnothing. \tag{4.8}$$

Comparing (4.7) and (4.8), we see that $f(\bigcup_{x \in T} U_x) \neq \bigcup_{x \in T} f(U_x)$, thus f is not union-subadditive. This is a contradiction since f is given to be union- and intersection-subadditive. Therefore, the claim of the lemma holds. □

Lemma 4.11. *For a finite non-empty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be a focus function that is both union- and intersection-subadditive. For every non-empty $T \subseteq S$, and for every $x \in T$, if $T \in focused(f)$ and $x \in special(f, T)$, then there exists a $y \notin T$ such that $y \in partner(f, x)$.*

Proof. Assume that there exists a $T \subseteq S$ and there exists an $x \in T$ such that $T \in focused(f)$ and $x \in special(f, T)$ and for every $y \notin T$, contrary to the claim, $y \in partner(f, x)$, i.e. $y \notin T$ but there exists $W_y \subseteq S$ such that $x \in W_y$ and $f(W_y) = \varnothing$. Following from this assumption are:

$$\text{for each } y \notin T, y \notin W_y \implies \bigcap_{y \notin T} W_y \subseteq T \tag{4.9}$$

$$\text{for each } y \notin T, x \in W_y \implies x \in \bigcap_{y \notin T} W_y \tag{4.10}$$

$$(4.9), (4.10), \text{ and } x \in special(f, T) \implies f(\bigcap_{y \notin T} W_y) = \{q\} \tag{4.11}$$

$$\text{for each } y \notin T, f(W_y) = \varnothing \implies \bigcup_{y \notin T} f(W_y) = \varnothing \tag{4.12}$$

Comparing (4.11) and (4.12), we see that $f(\bigcap_{y \notin T} W_y) \neq \bigcup_{y \notin T} f(W_y)$, thus f is not intersection-subadditive. This is a contradiction since f is given to be union- and intersection-subadditive. Therefore, the claim of the lemma holds. □

We are now ready to present and prove the following theorem, originally proved in [5], in our series of results which will eventually lead us to a generalized characterization theorem

for resource-minimal reaction systems.

Theorem 4.12. *For a finite non-empty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be a focus function. Then f is resource-minimal if and only if f is both union- and intersection-subadditive.*

Proof. First, we show that f is resource-minimal implies f is both union- and intersection-subadditive. Let $\mathcal{A} = (S, A)$ be the resource-minimal RS which defines f, and let $q \in S$ be the focus of f. For every non-empty $\mathcal{F} \subseteq \mathcal{P}(S)$, consider the claims $f(\bigcup \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$ and $f(\bigcap \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$. Since f is a focus function, there are two cases to consider for each claim: the value of f is either $\varnothing$ or $\{q\}$.

Claim 1. For every non-empty $\mathcal{F} \subseteq \mathcal{P}(S)$, $f(\bigcup \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$.

Case 1.1. $f(\bigcup \mathcal{F})) = \varnothing$. Since $\varnothing \subseteq \bigcup f(\mathcal{F})$, the claim holds.

Case 1.2. $f(\bigcup \mathcal{F})) = \{q\}$. Since f is defined by $\mathcal{A}$ and since $f = res_{\mathcal{A}} = \{q\}$, then there exists a reaction $a \in A$ such that $en_a(\bigcup \mathcal{F})$. Since $\mathcal{A}$ is resource-minimal, $|R_a| = 1$ and $|I_a| = 1$, so let $a = (\{x\}, \{y\}, \{q\})$. Thus, there exists a $T \subseteq \mathcal{F}$ such that $x \in T$ and $y \notin T$ and so $en_a(T)$. The product set $\{q\} = res_a(T)$ hence $q \in res_{\mathcal{A}}(T) = f(T)$. Since $f(T) \in \bigcup f(\mathcal{F})$, then $q \in \bigcup f(\mathcal{F})$ and so $f(\bigcup \mathcal{F}) = \{q\} \subseteq \bigcup f(\mathcal{F}) = \{q\}$ and the claim holds.

Since the claim holds for both cases, by Lemma 4.3 f is union-subadditive.

Claim 2. For every non-empty $\mathcal{F} \subseteq \mathcal{P}(S)$, $f(\bigcap \mathcal{F}) \subseteq \bigcup f(\mathcal{F})$.

Case 2.1. $f(\bigcap \mathcal{F}) = \varnothing$. Since $\varnothing \subseteq \bigcup f(\mathcal{F})$, the claim holds.

Case 2.2. $f(\bigcap \mathcal{F}) = \{q\}$. Since f is defined by $\mathcal{A}$ and since $f = res_{\mathcal{A}} = \{q\}$, then there exists a reaction $a \in A$ such that $en_a(\bigcap \mathcal{F})$. Since $\mathcal{A}$ is resource-minimal, $|R_a| = 1$ and $|I_a| = 1$, so let $a = (\{x\}, \{y\}, \{q\})$. Thus, there exists a $T \subseteq \mathcal{F}$ such that $x \in T$ and $y \notin T$ and $en_a(T)$. The product $\{q\} = res_a(T)$ hence $q \in res_{\mathcal{A}}(T) = f(T)$. Since $f(T) \in \bigcup f(\mathcal{F})$, then $q \in \bigcup f(\mathcal{F})$ and so $f(\bigcap \mathcal{F}) = \{q\} \subseteq \bigcup f(\mathcal{F}) = \{q\}$ and the claim holds.

Since the claim holds for both cases, by Lemma 4.3, f is intersection-subadditive. Since both claims hold for both cases, then f is both union- and intersection-subadditive.

We will now show that f is both union- and intersection-subadditive implies f is resource-minimal.

Let $q \in S$ be the focus of f. We construct a resource-minimal RS $\mathcal{A} = (S, A)$ such that $A = \{(\{x\}, \{y\}, \{q\}) : x, y \in S$ and there exists a $T \subseteq S$ such that $x \in special(f, T)$ and $y \in partner(f, x)\}$. Consider the claims $f \subseteq res_{\mathcal{A}}$ and $res_{\mathcal{A}} \subseteq f$. By proving these claims, we will show that $f = res_{\mathcal{A}}$ and thus f is resource-minimal. Since f is a focus function, there are two cases to consider for each claim: the value of f is either $\varnothing$ or $\{q\}$.

Claim 3. $f \subseteq res_{\mathcal{A}}$, i.e. for every $T \subseteq S, f(T) \subseteq res_{\mathcal{A}}(T)$.

Case 3.1. $f(T) = \varnothing$. The claim holds since $\varnothing \subseteq res_{\mathcal{A}}(T)$.

Case 3.2. $f(T) = \{q\}$. Then by Lemma 4.10 there exists an $x \in T$ such that $x \in special(f, T)$, and by Lemma 4.11 there exists a $y \notin T$ such that $y \in partner(f, x)$. Since $\mathcal{A}$ was constructed so that A contains reactions that meet these conditions, there exists a reaction $a \in A$ such that $a = (\{x\}, \{y\}, \{q\})$. Hence, $en_{\mathcal{A}}(T)$ and since $res_a = \{q\}$ then $q \in res_{\mathcal{A}}$. Thus, $f(T) \subseteq res_{\mathcal{A}}(T)$.

Claim 4. $res_{\mathcal{A}} \subseteq f$, i.e. for every $T \subseteq S, res_{\mathcal{A}}(T) \subseteq f(T)$.

Case 4.1. $res_{\mathcal{A}}(T) = \varnothing$. Since $\varnothing \subseteq f(T)$, the claim holds.

Case 4.1. $res_{\mathcal{A}}(T) = \{q\}$. Then there exists a reaction $a \in A$ such that $en_a(T)$ where $a = (\{x\}, \{y\}, \{q\})$ for some $x \in T$ and $y \notin T$. Also, $y \in partner(f, x)$ by the construction of $\mathcal{A}$. By the definition of f-partner, $f(T) \neq \varnothing$, but since f is a focus function with focus q, it must be that $f(T) = \{q\}$. Thus $res_{\mathcal{A}}(T) \subseteq f(T)$.

We have shown that $f \subseteq res_{\mathcal{A}}$ and $res_{\mathcal{A}} \subseteq f$, hence $f = res_{\mathcal{A}}$, i.e. f is defined by the RS $\mathcal{A}$ which was constructed to be resource-minimal. Thus, f is resource-minimal.

We have shown that f is resource-minimal implies f is both union- and intersection-subadditive and f is both union- and intersection-subadditive implies f is resource-minimal, therefore the proof is complete. □

Normal form and distributivity

The following two theorems are the last two results that we need to prove the generalized characterization theorem for arbitrary resource-minimal RS functions.

Definition 4.13. For a RS $\mathcal{A} = (S, A)$, the *"specialized entity" reaction system* dedicated to producing one specific entity, $q \in S$, is defined by:

$$\mathcal{A}^q = (S, A^q) \text{ such that } A^q = \{(R_a, I_a, \{q\} : a \in A \text{ and } q \in P_a\}. \tag{4.13}$$

Theorem 4.14 (Normal Form)**.** *Let $\mathcal{A} = (S, A)$ be a reaction system and $q \in S$. Let $\mathcal{A}^q = (S, A^q)$ where $A^q = \{(R_a, I_a, \{q\}) : a \in A \text{ and } q \in P_a\}$.*

1. *$\mathcal{A}$ is equivalent to $\bigcup_{q \in S} \mathcal{A}^q$.*
2. *$\mathcal{A}$ is resource-minimal if and only if $\mathcal{A}^q$ is resource-minimal for every $q \in S$.*

Proof. We prove this theorem through two claims. *Claim 1.* Let $\mathcal{A}' = \bigcup_{q \in S} \mathcal{A}^q$. We will show that,

$$res_{\mathcal{A}}(T) = res_{\mathcal{A}'}(T) \text{ for every } T \subseteq S. \tag{4.14}$$

For each $q \in S$, let $a_q \in A$, i.e. $a_q = (R_a, I_a, \{q\})$ such that $a \in A$ and $q \in P_a$. For each $T \subseteq S$ and for every $a \in A$, let $P_a = \{q_1, q_2, ..., q_n\}$ where $n = |P_a|$, thus,

$$res_a(T) = P_a = \{q_1, q_2, ..., q_n\} = \bigcup_{i=1}^{n} res_{a_q i}(T) = \bigcup_{q \in P_a} res_{a_q}(T). \tag{4.15}$$

Hence,

$$\bigcup_{a \in A} res_a(T) = \bigcup_{a \in A} \bigcup_{q \in P_a} res_{a_q}(T) \text{ for every } T \subseteq S. \tag{4.16}$$

So it follows that,

$$\text{for every } T \subseteq S, res_{\mathcal{A}}(T) = res_{\mathcal{A}'}(T). \tag{4.17}$$

Therefore $\mathcal{A}$ is functionally equivalent to $\bigcup_{q \in S} \mathcal{A}^q$.

Claim 2. First we will show that $\mathcal{A}$ is resource-minimal implies $\mathcal{A}^q$ is resource-minimal for each $q \in S$. Since $\mathcal{A}$ is resource-minimal, then for each reaction $a \in A$, $a = (R_a, I_a, P_a)$. Since $\mathcal{A}$ is resource-minimal, then $|R_a| = 1$ and $|I_a| = 1$. Since each reaction $a_q \in \mathcal{A}^q$ was constructed such that $a_q = (R_a, I_a, \{q\})$ for some $a \in A$, then a_q is resource-minimal. Therefore $\mathcal{A}^q$ is resource-minimal.

Next we will show that all $\mathcal{A}^q \in \mathcal{A}'$ are resource-minimal implies $\mathcal{A}$ is resource-minimal. Since each reaction $a_q = (R_a, I_a, \{q\})$ such that the reaction $a \in A$ is (R_a, I_a, P_a) then since a_q is resource-minimal, then $|R_a| = |I_a| = 1$, hence a is resource-minimal. From Claim 1, we know $\mathcal{A} \sim \bigcup_{q \in S} \mathcal{A}^q$ hence all $a \in A$ have an equivalent set of reactions in $\bigcup_{q \in S} \mathcal{A}^q$. Therefore, $\mathcal{A}$ is resource-minimal. □

Distributivity

Definition 4.15. Given an RS function $f : \mathcal{P}(S) \to \mathcal{P}(S)$ and $q \in S$, the *specialized entity q-component of f* is defined by:

$$f^q(T) = f(T) \cap \{q\} \text{ for every } T \subseteq S.$$

Theorem 4.16 (Distributivity)**.** *Let f be an RS function over S. For every $q \in S$:*

1. *f is union-subadditive if and only if each f^q is union-subadditive.*
2. *f is intersection-subadditive if and only if each f^q is intersection-subadditive.*

Proof. We prove this theorem through two claims. *Claim 1.* Let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ and $q \in S$.

First we will show that f is union-subadditive implies each f^q is union-subadditive. Suppose f is union-subadditive, then for every $X, Y \subseteq S$:

$$f(X \cup Y) \subseteq f(X) \cup f(Y) \tag{4.18}$$

Hence,

$$f(X \cup Y) \cap \{q\} \subseteq \big(f(X) \cup f(Y)\big) \cap \{q\} = \big(f(X) \cap \{q\}\big) \cup \big(f(Y)) \cap \{q\}\big). \tag{4.19}$$

By applying Definition 4.15 to the three sets $X \cup Y$, X, and Y, we have:

$$\begin{aligned} f^q(X \cup Y) &= f(X \cup Y) \cap \{q\} \\ f^q(X) &= f(X) \cap \{q\} \\ f^q(Y) &= f(Y) \cap \{q\} \end{aligned} \tag{4.20}$$

Substituting the left sides of Equations 4.20 into 4.19, we have shown that f^q is union-subadditive:

$$f^q(X \cup Y) \subseteq f^q(X) \cup f^q(Y). \tag{4.21}$$

Next we will show that all f^q are union-subadditive implies f is union-subadditive. Suppose f^q is union-subadditive. Then for every $X, Y \subseteq S$:

$$f^q(X \cup Y) \subseteq f^q(X) \cup f^q(Y) \tag{4.22}$$

which hold for every $q \in S$, hence

$$\bigcup_{q \in S} f^q(X \cup Y) \subseteq \bigcup_{q \in S} f^q(X) \cup \bigcup_{q \in S} f^q(Y). \tag{4.23}$$

Since $f^q(X) = \{q\}$ if $q \in X$ (by definition of f^q), then for every $W \subseteq S$

$$f(W) = \bigcup_{q \in S} f^q(W). \tag{4.24}$$

Thus, by applying this identity for $X \cup Y, X,$ and Y and substituting into 4.23, we see that f is union-subadditive:

$$f(X \cup Y) \subseteq f(X) \cup f(Y). \tag{4.25}$$

Claim 2. The proof is identical to the first part except we refer to intersection-subadditive in place of union-subadditivity, and we apply the definition for intersection-subadditive in place of that for union-subadditivity. □

Resource-minimal functions

Using the last three theorems, the generalized characterization theorem for arbitrary resource-minimal reaction systems can now be proved.

Theorem 4.17. *For a finite nonempty set S, let $f : \mathcal{P}(S) \to \mathcal{P}(S)$ be an reaction system function. Then f is resource-minimal if and only if f is both union- and intersection-subadditive.*

Proof. First, we will show that f is resource-minimal implies f is both union- and intersection-subadditive. Suppose f is a resource-minimal RS functions and let $\mathcal{A} = (S, A)$ be the resource-minimal RS defining f. Then by Theorem 4.14:

$$\mathcal{A} \sim \bigcup_{q \in S} \mathcal{A}^q$$

Hence

$$res_{\mathcal{A}} = \bigcup_{q \in S} res_{\mathcal{A}^q}.$$

We know that f_q is a focus function by Definition 4.9, and from (4.24) we know that for every $q \in S$:

$$f = \bigcup_{q \in S} f^q.$$

Then by Theorem 4.12, f_q is both union- and intersection-subadditive. Thus, by Theorem 4.16, f is both union- and intersection-subadditive.

Next we will show that f is both union- and intersection-subadditive implies f is resource-minimal. Suppose f is both union- and intersection-subadditive. Then by Theorem 4.16, for every $q \in S$, f^q is also both union- and intersection-subadditive. Also, by Definition 4.9, f^q is a focus function. Hence by Theorem 4.12, each f^q is resource-minimal. For each $q \in S$, let $\mathcal{A}^q$ be the resource-minimal RS which defines f^q, i.e.:

$$f^q = res_{\mathcal{A}^q}.$$

Applying this identity to (4.24), we have

$$f = \bigcup_{q \in S} f^q = \bigcup_{q \in S} res_{\mathcal{A}^q}.$$

Let $\mathcal{A} = \bigcup_{q \in S} \mathcal{A}^q$, then $res_{\mathcal{A}} = \bigcup_{q \in S} res_{\mathcal{A}^q}$. Also, $\mathcal{A} = (S, \bigcup_{q \in S} A^q)$, and since each reaction set A^q is resource-minimal, then so is $\bigcup_{q \in S} A^q$. Hence $\mathcal{A}$ is resource-minimal as is $res_{\mathcal{A}}$.

Thus $f = res_{\mathcal{A}}$ is resource-minimal.

We have shown that f is resource-minimal implies f is both union- and intersection-subadditive and f is both union- and intersection-subadditive implies f is resource-minimal, therefore the proof is complete. □

The RS function mentioned in the introduction to Section 4.3.3, defined by the many reactions in Table 3.5, is both union- and intersection-subadditive. Theorem 4.17 tells us that a functionally equivalent resource-minimal RS exists. In fact, the reactions shown below is the set of resource-minimal reactions which defines that functionally equivalent RS.

$$\begin{aligned} a_1 = &\quad \{b\}, \{c\}, \{b\}) \\ a_2 = &\quad (\{a\}, \{b\}, \{c\}) \\ a_3 = &\quad (\{a\}, \{c\}, \{b\}) \\ a_4 = &\quad (\{b\}, \{a\}, \{c, d\}) \end{aligned}$$

How the non-minimal RS was minimized to obtain this resource-minimal one is discussed in Chapter 5.

4.3.4 Product-minimal

As mentioned earlier, a resource-minimal RS where every reaction has a product set with cardinality 1 is called product-minimal as defined in [19]. There is no need for an additional characterization theorem to determine if an RS function is product-minimal since a functionally equivalent RS can be easily obtained for *any* resource-minimal RS that is not product-minimal. For example, the reaction $(\{1\}, \{2\}, \{3, 4\})$ is equivalent to these two reactions together: $(\{1\}, \{2\}, \{3\})$ and $(\{1\}, \{2\}, \{4\})$.

The idea behind Proposition 4.18 below is very similar to that of Theorem 4.14, but the statement of this proposition is clearer for the following purpose. A reaction with more than one product entity can be split into multiple reactions. RS Tools uses this fact when it is minimizing the number of Boolean functions for an RS.

Proposition 4.18. *Let $a = (R_a, I_a, P_a)$ be a reaction such that $|P_a| > 1$. There exists a set of reactions $\{a'_1, a'_2, ..., a'_n\}$ such that $res_a = res_{a'_1} \cup res_{a'_2} \cup ... \cup res_{a'_n}$.*

Proof. Let reaction $a = (R_a, I_a, P_a)$ such that $|P_a| = n > 1$. Let $P_a = \{p_1, p_2, ..., p_n\}$. Construct a set of n reactions: $\{a'_1, a'_2, ..., a'_n\}$ such that $a'_i = (R_a, I_a, \{p_i\}), i = 1, 2, ..., n$. Hence,

$$\bigcup_{i=1}^{n} res_{a'_i} = res_{a'_1} \cup res_{a'_2} \cup ... \cup res_{a'_n}.$$

If a is not enabled, then neither are any of the reactions a'_i since they have the same reactant and inhibitor sets. Thus,

$$\begin{aligned}\bigcup_{i=1}^{n} res_{a'_i} &= res_{a'_1} \cup res_{a'_2} \cup ... \cup res_{a'_n} \\ &= \varnothing \cup \varnothing \cup ... \cup \varnothing \\ &= \varnothing = res_a.\end{aligned}$$

If a is enabled, then

$$\begin{aligned}\bigcup_{i=1}^{n} res_{a'_i} &= res_{a'_1} \cup ... \cup res_{a'_n} \\ &= \{p_1\} \cup ... \cup \{p_n\} \\ &= \{p_1, p_2, ..., p_n\} \\ &= P_a \\ &= res_a.\end{aligned}$$

Therefore, $res_a = \bigcup_{i=1}^{n} res_{a'_i}$. □

Proposition 4.19. *For a set of reactions $a_1 = (R_a, I_a, P_1),\ ...\ , a_n = (R_a, I_a, P_n)$ such that every reaction has the same reactant and inhibitor sets and $P_1, ..., P_n$ are products sets not necessarily singletons, then this set of reactions is functionally equivalent to $a = (R_a, I_a, P_a)$ where $P_a = P_1 \cup ... \cup P_n$.*

The proof of Proposition 4.19 follows the argument in the proof of Proposition 4.18 in reverse order. Even though product-minimal reactions are defined in some papers as also having the resource-minimal property, both propositions apply to any (r, i) class of reactions.

4.4 Comparison of minimal, almost minimal, and unrestricted reaction systems

4.4.1 Complexity of decision problems

In Section 2.4, (r, i) classes of reaction systems were defined. Although reaction systems containing only $(1, 1)$ reactions (which involve exactly two resources) are referred to as resource-minimal (most papers refer to them as "minimal" reaction systems), reaction systems with reactions involving at most three resources (i.e., $(1, 2)$ and $(2, 1)$ reactions) are referred to as *almost minimal* reaction systems.

Some of the problems of interest for reaction systems are decidability problems for

occurrence, convergence, inverse function, and totality, which are discussed in [18] and [3]. Given a reaction system with a purposely specific set of reactions, it is often important to know the reachability of a specific result. Moreover, it may be desirable to know if the result can be obtained after a number of steps. This problem, and others discussed below, are definitively decidable due to the fact that the background set is finite (see [18]) and so the discussion below focuses on the complexity classes of these decidability problems.

In [18], the following comparisons of complexity of decidability problems are made between resource-minimal and almost minimal reaction systems. Many of the problems are of low polynomial complexity for resource-minimal reaction systems, but for almost minimal systems the same problems are NP- or co-NP-complete. In [18], this difference in complexity is described as comparable to the difference between 2-SAT and 3-SAT, and it is conjectured that the gap in complexity between resource-minimal and almost minimal reaction systems is greater than the gap between almost minimal and unrestricted reaction systems.

The occurrence and convergence decision problems for reaction systems are defined in [9], and in [18]. Sometimes both of these problems are combined as one definition referred to as the convergence problem; the convergence problems extends the occurrence problem.

Definition 4.20 (Occurrence and convergence decision problems)**.**
For a given RS $\mathcal{A} = (S, A)$ and an integer $n \geq 1$:

1. The occurrence problem is to determine if an element $x \in S$ occurs at the n^{th} step of at least one sequence generated by $\mathcal{A}$ which has at least n steps.
2. The convergence problem is to determine if x occurs at the n^{th} step of *every* sequence generated by $\mathcal{A}$ which has at least n steps.

It is proved in [18] that the (combined) convergence problem is co-NP-complete for almost minimal reaction systems. To compare with minimal reaction systems, in [9] it is proved that the occurrence problem is NP-complete and the convergence problem is co-NP-complete for minimal reaction systems. For unrestricted reaction systems, it is proved in [9] that both of the problems are PSPACE-complete.

The inverse function problems for reaction systems is defined in [18].

Definition 4.21 (Inverse problem)**.** Given a RS $\mathcal{A} = (S, A)$ and $T \subseteq S$, the problem is to determine if there exists $U \subseteq S$ such that $f_{\mathcal{A}}(U) = T$.

It is proved in [18] that the inverse problem is NP-complete for almost minimal reaction systems. It is stated in [18] that there is no corresponding result for resource-minimal reaction systems [that prove a lesser complexity], however [17] shows that this problem is NP-complete for general reaction systems. We can still compare resource-minimal and general reaction

systems for this problem considering that technically resource-minimal reaction systems are also almost minimal reaction system by definition.

Other decision problems of interest for reaction systems are the totality problem and the problem concerning termination-freeness, and although their complexity has not been compared between minimal and almost minimal reaction systems, they are proved in [17] to be co-NP-complete for reaction systems in general.

4.4.2 Sequence-generating capabilities

In Section 3.1.4, we discussed the lower bound for the capability of certain non-minimal classes of reaction systems to generate long sequences (terminating and cyclic). Now that we have defined resource-minimal and almost minimal reaction systems in this chapter, we can compare their sequence-generating capabilities with the higher-class reaction systems.

This section shows that $(1,1),(1,2)$, and $(2,1)$ RS functions have the capability to generate terminating sequences and cycles of any length, i.e. these functions exists. For a given RS, there is still a bound on these lengths (see Theorem 3.4). The following theorems are discussed and proved in [15] and [16] and [18]. The first two theorems below show the lower bound of the capability of resource-minimal reaction systems to generate sequences depending on the cardinality of the background set. Theorem 4.23 is described in [18] as the best known lower bound for the length of terminating sequences able to be generated by resource-minimal reaction systems.

Theorem 4.22. *For all $k \geq 1$, there is a $(1,1)$ reaction system having $3k$ elements in the background set and having a terminating state sequence of length $4 \cdot 2^k - 3$, as well as a $(1,1)$ reaction system having a background set of length $3k+3$ and having a cycle of length $4 \cdot 2^k - 1$.*

The proof of Theorem 4.22 in [16] begins with the construction of a reaction system, assumed to have a terminating state sequence of length $k \geq 1$, which has a terminating state sequence of length $2k+3$. The proof uses this result to set up two steps of a linear recursion problem and concludes that the solution gives the lengths shown in Theorem 4.22. The proof of Theorem 4.23, also in [16], has a similar approach.

Theorem 4.23. *For every $k \geq 1$, there is a $(1,1)$ reaction system having $4k$ elements in the background set and having a terminating state sequence of length m_k where $m_1 = 7, m_2 = 26, m_k = 3 \cdot 3^k + 9(3^{k-3} - 1)/2 + 2, k \geq 3$.*

According to [18], certain functions not obtainable by resource-minimal reaction systems can be obtained using almost minimal ones. The next two theorems show that we can obtain

terminating state sequences and cycles that are exponentially longer than the ones whose existence is described in the previous two theorems.

Theorem 4.24. *For every $k \geq 1$, there is an almost minimal [i.e., $(1,2) or (2,1)$] reaction system having $3k$ elements in the background set and a terminating state sequence of length $(8/3) \cdot 3^k - 1$.*

Theorem 4.25. *For every $k \geq 1$, there is an almost minimal [i.e., $(1,2)$ or $(2,1)$] reaction system having $3k+2$ elements in the background set and having a cycle of length $(8/3) \cdot 3^k$.*

The proofs of these last two theorems are in [18], and as with the previous two theorems from [16], the approach is similar: from the result of a construction proof, a recurrence problem is set up the solution to which is the conclusion.

5. Minimizing reaction systems

This chapter discusses two methods of minimizing reaction systems. The first method uses Boolean logic minimization techniques to minimize a RS. If the RS is functionally equivalent to a $(1, 1)$ RS, the method can find it. The second method does not find a functionally equivalent (by definition) $(1, 1)$ RS, but it can derive a $(1, 1)$ RS, for any RS, that "simulates" the RS.

5.1 Logic minimization applied to reaction systems

As we have seen in previous sections, a reaction system can be functionally equivalent to a reaction system that is simpler in one or more of the following ways. It can have a smaller cardinality of reactions. It can have a smaller cardinality of resources utilized by the evaluation of the res function for the RS as a whole. Its class can be smaller, meaning that the cardinality of resources required for each and every reaction is less. Its Boolean vector function can be simpler with fewer Boolean functions and/or fewer terms.

By minimizing a reaction system, we can try to find a simpler reaction system. Theorem 4.17 tells us under what circumstances (regarding subadditive properties) a given RS is functionally equivalent to a $(1, 1)$ RS, but according to [18], there is no known corresponding characterization theorem for any other specific classes of reaction systems. In any case, algorithms exist, which pre-date the introduction of the theoretical reaction systems model, which can be used to find the minimized functionally equivalent RS.

In Section 3.3, it is shown that a RS function can be represented by a Boolean vector function. Each Boolean function in the vector function determines whether or not a certain entity is produced, but when the product of a reaction includes more than one entity, it would be more efficient to represent such a reaction with one Boolean function instead of many. Furthermore, if multiple reactions produce the same product set, all such reactions can be represented with one Boolean function instead of many. Further still, we might discover that some reactions are redundant, or we might find that some Boolean functions are equivalent to ones with fewer terms or literals. So we see that there may be plenty of opportunities for minimization algorithms to minimize a RS represented as Boolean functions.

There are many algorithms for minimizing logic formulas, as mentioned in Section 3.3. One of the most efficient and flexible algorithms is Espresso which is widely used in both academia and industry. Espresso is available as a command-line program for most computing platforms. The algorithm and the details of how it works have been widely discussed, such

as in [2] in which the objective of Espresso is stated as being to minimize the number of product terms and literals.

The Espresso program includes an algorithm to find the guaranteed minimal number of product terms while heuristically minimizing the number of literals, and this mode of operation is described in the Espresso documentation as being potentially computationally time-consuming [14].

Another option in Espresso is an algorithm which is heuristic both in minimizing product terms and literals. In this mode, Espresso finds a solution in dramatically shorter execution time. It is not guaranteed to find the optimal minimization, but for small functions the solutions are nearly always optimal, otherwise Espresso finds a solution that is very close to the optimal solution. In any case, when using Espresso as part of minimizing a reaction system function, the outcome is always functionally equivalent.

Example 5.1.1. The RS shown in Table 3.5 is presented as its equivalent Boolean vector function $F(x)$ in Table 3.6. To minimize the RS, we minimize the Boolean functions and then convert those minimal functions back into an RS. We will use the Espresso heuristic algorithm to do the logic minimization. Espresso requires a truth table as input, and the full truth table with multiple inputs and outputs for this example is included within Table 3.7, however the table includes rows where $F(x) = \vec{0}$ (i.e., $res_{\mathcal{A}} = \varnothing$). Espresso does not need those rows so they can be excluded from the input to Espresso. The remaining rows are correspond to enabled reactions.

Using RS Tools to prepare the input for and interpret the output from Espresso (i.e., RS Tools is a visual application which interfaces with Espresso which is a back-end tool), RS Tools reports that the minimized Boolean functions are those shown in Table 5.1. Furthermore, RS Tools converts these Boolean functions into the equivalent minimized RS which is shown in Table 5.2.

Table 5.1: Minimization of Boolean vector function shown in Table 3.6.

$$
\begin{aligned}
G(X) = &\quad \Big[g_1(X) \quad g_2(X) \quad g_3(X) \Big] \\
&\text{where:} \\
g_1(X) = &\quad (x_1 \wedge \overline{x}_3) \vee (x_2 \wedge \overline{x}_3) \\
g_2(X) = &\quad (x_2 \wedge \overline{x}_1) \\
g_3(X) = &\quad (x_1 \wedge \overline{x}_2)
\end{aligned}
$$

Table 5.2: The minimal RS $\mathcal{B}$ which is functionally equivalent to RS $\mathcal{A}$.

	$f_1 = (a \wedge \overline{c}) \vee (b \wedge \overline{c})$
	$f_2 = (b \wedge \overline{a})$
$b_1 = (\{a\}, \{b\}, \{c\})$	$f_3 = (a \wedge \overline{b})$
$b_2 = (\{a\}, \{c\}, \{b\})$	where the product sets associated with each fn are:
$b_3 = (\{b\}, \{a\}, \{c, d\})$	$f_1 : \{b\}$
$b_4 = (\{b\}, \{c\}, \{b\})$	$f_2 : \{c, d\}$
	$f_3 : \{c\}$

In summary, given a reaction system $\mathcal{A} = (S, A)$, RS Tools generates a truth table with a row for every subset of S that is enabled by $\mathcal{A}$. The truth table is processed by Espresso which returns another truth table which is converted by RS Tools into the equivalent minimized reaction system $\mathcal{B}$.

5.1.1 From state transition map to minimized RS

Suppose all of the possible states of a state transition system have been observed and mapped to produce the state transition map shown in Figure 5.1. The map can also be interpreted as a one-out directed graph with vertex labels that correspond to the states represented in the map. We will convert this mapping into a minimized reaction system $\mathcal{A} = (S, A)$ that defines a total function, as defined in [7], i.e. $res_{\mathcal{A}} : \mathcal{P}(S) \to \mathcal{P}(S)$ and $res_{\mathcal{A}}$ is defined for every proper subset of S.

The following definition and proposition proved in [10] can be used to verify that Figure 5.1 is the 0-context graph of some RS. Then we will derive an RS for this map, and finally we will minimize the RS.

Definition 5.1. For a RS $\mathcal{A}$, the *0-context graph* of $\mathcal{A}$ is the one-out graph (i.e. there is exactly one outgoing edge from each vertex) $G^0_{\mathcal{A}} = (\mathcal{P}(S), E)$ with edge set $E = \{(v, res_{\mathcal{A}}(v)) \,|v \in \mathcal{P}(S)\}$.

Proposition 5.2. *A one-out graph G with vertex set $\mathcal{P}(S)$ is a 0-context graph of a RS if and only if $(\varnothing, \varnothing)$ and $(S, \varnothing)$ are edges of G.*

Proof. Let $\mathcal{A} = (S, A)$ be a RS and graph $G = (\mathcal{P}(S), E)$ be a one-out graph. We will show:

1. If G is a 0-context graph of $\mathcal{A}$ then $(\varnothing, \varnothing), (S, \varnothing) \in E$.
2. If $(\varnothing, \varnothing), (S, \varnothing) \in E$ then G is a 0-context graph of $\mathcal{A}$.

Claim 1. Assume G is a 0-context graph of $\mathcal{A}$, then $E = \{(V, res_{\mathcal{A}}(V)) : V \in \mathcal{P}(S)\}$. Since both $\varnothing, S \in \mathcal{P}(S)$ and $res_{\mathcal{A}}(\varnothing) = res_{\mathcal{A}}(S) = \varnothing$ for any RS, then edges $(\varnothing, res_{\mathcal{A}}(\varnothing)) = (\varnothing, \varnothing)$

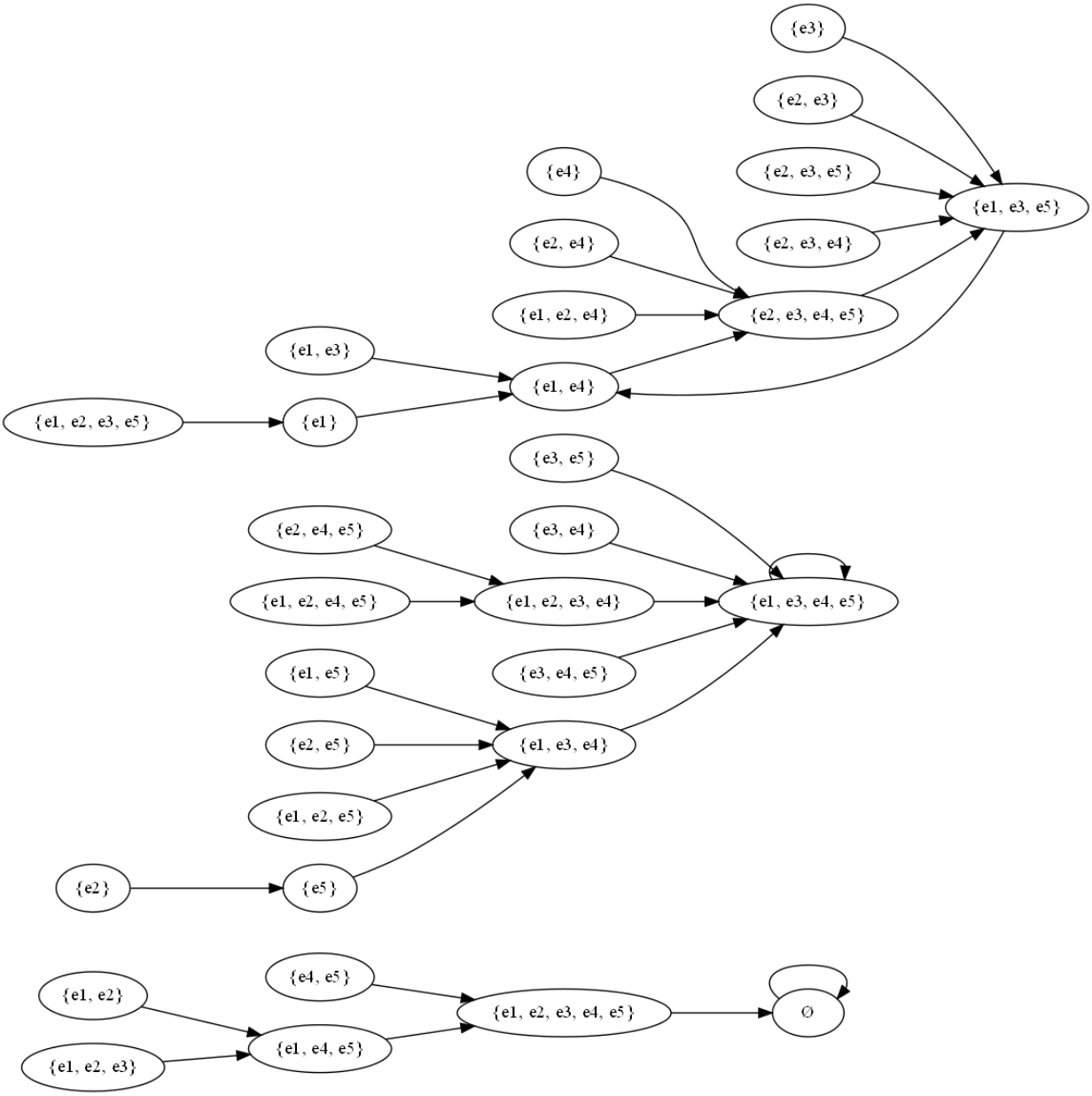

Figure 5.1: State transition mapping.

and $(S, res_{\mathcal{A}}(S)) = (S, \varnothing)$ are in E.

Claim 2. Assume $(\varnothing, \varnothing), (S, \varnothing) \in E$. Since G is one-out, there is exactly one outgoing edge from each vertex of G. Define reaction set:

$$A = \{(V, S\backslash V, U) : V \in \mathcal{P}(S), V \notin \{\varnothing, S\}, U \in \mathcal{P}(S) \text{ and } U \text{ is unique}\}.$$

Since $V \cap (S\backslash V) = \varnothing$, every reaction in A is enabled. Since for each $V \in \mathcal{P}(S) - \{0, S\}$, the edge $(V, res_{\mathcal{A}}(V))$ is in E and is unique and since $(\varnothing, \varnothing), (S, \varnothing) \in E$, then

$$E = \{(V, res_{\mathcal{A}}(V)) : V \in \mathcal{P}(S)\}.$$

Thus, by definition, G is a 0-context graph of $\mathcal{A}$.

□

From the map, we can see that the union of the sets used to label the states is $S = \{e1, e2, e3, e4, e5\}$. After a mildly tedious examination of the map, we can see that all elements of $\mathcal{P}(S)$ are represented as vertices. We also must note that the map is a one-out graph and that the edges $(\varnothing, \varnothing)$ and $(S, \varnothing)$ are present. Hence, by Proposition 5.2 and definition of total, the map is a 0-context graph of some RS that defines a total function.

Now we will rely on Definition 3.3 and the following proposition from [11] to construct a RS from the map.

Proposition 5.3. *Let S be a finite set. For every function $f : \mathcal{P}(S) \to \mathcal{P}(S)$, there exists a reaction system $\mathcal{A} = (S, A)$ such that $f = res_{\mathcal{A}}$.*

To paraphrase the proof of the proposition and relate it to the construction of a RS for the map, we need to construct a RS over S for which all of the reactions in set A are maximally inhibited. In particular, for each edge (V_1, V_2), where $V_1, V_2 \subseteq S$, we construction a reaction $(V_1, S\backslash V_1, V_2)$.

The maximally inhibited reaction system constructed in this manner is shown in Table 5.3. Effectively there are 30 reactions in the reaction set since there are no such reactions $(\varnothing, \varnothing, P_a)$ and $(S, \varnothing, P_a)$.

Although it is not necessary to check if a reaction system is subadditive before minimizing it, this reaction system was verified by RS Tools to be union- and inter-section subadditive. So there exists a functionally equivalent resource-minimal RS.

After minimizing this RS using Espresso via RS Tools, the minimized RS has the reaction set shown in Table 5.4. As we expect, the minimized RS is resource-minimal, and the cardinality of the reaction set has been simplified from 30 down to 10. More details of how this example was minimized are shown in Appendix A.1.

Table 5.3: Maximally inhibited RS for the map in Figure 5.1.

$a_1 = (\emptyset, \{e5, e1, e2, e4, e3\}, \emptyset)$ (see *)
$a_2 = (\{e5\}, \{e1, e2, e4, e3\}, \{e1, e3, e4\})$
$a_3 = (\{e1\}, \{e5, e2, e4, e3\}, \{e1, e4\})$
$a_4 = (\{e2\}, \{e5, e1, e4, e3\}, \{e5\})$
$a_5 = (\{e4\}, \{e5, e1, e2, e3\}, \{e2, e3, e4, e5\})$
$a_6 = (\{e3\}, \{e5, e1, e2, e4\}, \{e1, e3, e5\})$
$a_7 = (\{e1, e5\}, \{e2, e4, e3\}, \{e1, e3, e4\})$
$a_8 = (\{e2, e5\}, \{e1, e4, e3\}, \{e1, e3, e4\})$
$a_9 = (\{e4, e5\}, \{e1, e2, e3\}, \{e1, e2, e3, e4, e5\})$
$a_{10} = (\{e3, e5\}, \{e1, e2, e4\}, \{e1, e3, e4, e5\})$
$a_{11} = (\{e1, e2\}, \{e5, e4, e3\}, \{e1, e4, e5\})$
$a_{12} = (\{e1, e4\}, \{e5, e2, e3\}, \{e2, e3, e4, e5\})$
$a_{13} = (\{e1, e3\}, \{e5, e2, e4\}, \{e1, e4\})$
$a_{14} = (\{e2, e4\}, \{e5, e1, e3\}, \{e2, e3, e4, e5\})$
$a_{15} = (\{e2, e3\}, \{e5, e1, e4\}, \{e1, e3, e5\})$
$a_{16} = (\{e3, e4\}, \{e5, e1, e2\}, \{e1, e3, e4, e5\})$
$a_{17} = (\{e1, e2, e5\}, \{e4, e3\}, \{e1, e3, e4\})$
$a_{18} = (\{e1, e4, e5\}, \{e2, e3\}, \{e1, e2, e3, e4, e5\})$
$a_{19} = (\{e1, e3, e5\}, \{e2, e4\}, \{e1, e4\})$
$a_{20} = (\{e2, e4, e5\}, \{e1, e3\}, \{e1, e2, e3, e4\})$
$a_{21} = (\{e2, e3, e5\}, \{e1, e4\}, \{e1, e3, e5\})$
$a_{22} = (\{e3, e4, e5\}, \{e1, e2\}, \{e1, e3, e4, e5\})$
$a_{23} = (\{e1, e2, e4\}, \{e5, e3\}, \{e2, e3, e4, e5\})$
$a_{24} = (\{e1, e2, e3\}, \{e5, e4\}, \{e1, e4, e5\})$
$a_{25} = (\{e1, e3, e4\}, \{e5, e2\}, \{e1, e3, e4, e5\})$
$a_{26} = (\{e2, e3, e4\}, \{e5, e1\}, \{e1, e3, e5\})$
$a_{27} = (\{e1, e2, e4, e5\}, \{e3\}, \{e1, e2, e3, e4\})$
$a_{28} = (\{e1, e2, e3, e5\}, \{e4\}, \{e1\})$
$a_{29} = (\{e1, e3, e4, e5\}, \{e2\}, \{e1, e3, e4, e5\})$
$a_{30} = (\{e2, e3, e4, e5\}, \{e1\}, \{e1, e3, e5\})$
$a_{31} = (\{e1, e2, e3, e4\}, \{e5\}, \{e1, e3, e4, e5\})$
$a_{32} = (\{e1, e2, e3, e4, e5\}, \emptyset, \emptyset)$ (see *)

* The reactions $a_1 = (\varnothing, S, \varnothing)$ and $a_{32} = (S, \varnothing, \varnothing)$ are not reactions normally included in a RS reaction set, but they are shown here so that we include reactions corresponding to every state transition in Figure 5.1. So this reaction system effectively has 30 reactions.

Table 5.4: Resource-minimal RS for the map in Figure 5.1.

$a_1 = (\{e3\}, \{e1\}, \{e5, e1, e3\})$
$a_2 = (\{e3\}, \{e2\}, \{e1\})$
$a_3 = (\{e1\}, \{e4\}, \{e1\})$
$a_4 = (\{e3\}, \{e5\}, \{e1\})$
$a_5 = (\{e5\}, \{e3\}, \{e4, e3, e1\})$
$a_6 = (\{e4\}, \{e3\}, \{e2, e4\})$
$a_7 = (\{e4\}, \{e5\}, \{e3\})$
$a_8 = (\{e4\}, \{e2\}, \{e3, e5, e4\})$
$a_9 = (\{e1\}, \{e5\}, \{e4\})$
$a_{10} = (\{e5\}, \{e2\}, \{e4\})$

Even if a reaction system is not subadditive, Espresso will typically minimize a maximally inhibited RS to a potentially much smaller RS.

5.2 Reaction systems simulated by resource-minimal ones

Suppose a given RS $\mathcal{A}$ is known to not be functionally equivalent to a $(1, 1)$ RS. If we still want to have a $(1, 1)$ RS $\mathcal{B}$ that can generate the same sequence as $\mathcal{A}$ and we are willing to obtain the sequence by taking each step from $res^n_{\mathcal{B}}$, where $n \geq 2$, then we can construct $\mathcal{B}$ from $\mathcal{A}$ using methods described in this section. Only construction methods for $n = 2$ and $n = 3$ are discussed in this section. Once constructed, $\mathcal{B}$ "simulates" $\mathcal{A}$. The term "simulate" in this section does not refer to simulation in computer software although this

latter type of simulation, a completely different context, is discussed in Chapter 6.

5.2.1 Constructing a minimal RS to simulate an arbitrary RS

Let f be a function defined by an arbitrary reaction system $\mathcal{A}$. If f does not meet the conditions of Theorem 4.17, then f is not resource-minimal. In [20] is detailed a method for constructing a resource-minimal RS, $\mathcal{A}_M$ from $\mathcal{A}$ in such a way that we can still, systematically extract the same result sequence that we would get from $f_{\mathcal{A}}$. As one might expect, there is a price to be paid. Although $\mathcal{A}_M$ may be more efficient in terms of the number of resources used per reaction, the trade-off is that whereas $f_{\mathcal{A}}(T) = res_{\mathcal{A}}(T)$, i.e. one step, $f_{\mathcal{A}}(T) = f^3_{\mathcal{A}_M} = res_{\mathcal{A}_M}(res_{\mathcal{A}_M}(res_{\mathcal{A}_M}(T)))$, i.e. a three-step process is required.

Theorem 5.4. *For any reaction system $\mathcal{A}$ (over the background set S), there is a resource-minimal reaction system $\mathcal{A}_M$ (with a background set containing S) such that, for all $T \in \mathcal{P}(S)\backslash\{\varnothing, S\}$, $f_{\mathcal{A}}(T) = f^3_{\mathcal{A}_M}(T)$. Thus, the function value $f_{\mathcal{A}}(T)$ appears as the third step in the sequence of $\mathcal{A}_M$ starting with T.*

Proof. From the given RS $\mathcal{A} = (S, A)$ we construct a resource-minimal reaction system $\mathcal{A}_M = (S', A')$ such that $S \subset S'$ and $f_{\mathcal{A}}(T) = f^3_{\mathcal{A}_M}(T)$.

For each reaction $a_i \in A, i = 1, 2, ..., |A|$, define a new entity ρ_i. (Note: In [20] and [9], this new entity is considered the reaction itself being used as the name for the new entity.)

For each $x \in S$, define a new entity $\overline{x}$ as a "barred version" of x (not the complement of x), and let $\overline{S} = \{\overline{x} : x \in S\}$. Define three new entities not already in $S \cup \overline{S}$. Here we will use $B, \$,$ and E. Let $S' = S \cup \overline{S} \cup \{B, \$, E\}$. For every entity $x \in S$, the following reactions are included in the reaction set A'.

Set 1. $(\{x\}, \{\$\}, \{\overline{x}, B\})$

Set 2. Let $P_x = \{\rho_i : x$ is in the reactant set of the reaction $a_i\}$ be a product set of entities. If $P_x \neq \varnothing$, then we include the reaction $(\{B\}, \{\overline{x}\}, P_x)$ in A', otherwise no reactions are added to A' at this step.

Set 3. Let $I_x = \{\rho_i : x$ is in the inhibitor set of the reaction $a_i\}$ be a product set of entities. If $I_x \neq \varnothing$, then we include the reaction $(\{\overline{x}\}, \{\$\}, I_x)$ in A', otherwise no reactions are added to A' at this step.

Set 4. $(\{B\}, \{\$\}, \{E\})$

Set 5. $(\{E\}, \{\rho_i\}, \{P_i\}), i = 1, 2, ..., |A|$.

All of the reactions that are included in A' are resource-minimal, hence $\mathcal{A}_M$ is also resource-minimal. What remains is to justify that indeed $f_{\mathcal{A}}(T) = f^3_{\mathcal{A}_M}(T)$.

Let $W_0 \in S$ and consider $res_{\mathcal{A}_M}(W_0)$. If $W_0 \in \{\varnothing, S\}$, then neither $\mathcal{A}$ nor $\mathcal{A}_M$ are enabled and no sequence is generated. Consider the first step of the sequence generated by $\mathcal{A}_M(W_0)$. Only the reactions in Set 1 are enabled and so $W_1 = res_{\mathcal{A}_M}(W_0) = \{B\} \cup \{\overline{x} : x \in W_0\}$.

Next, consider $res_{\mathcal{A}_M}(W_1)$. Set 1 is not applicable, but Sets 2, 3, and 4 are. The product sets from Sets 2 and 3 contain the names of the reactions that are not enabled for T in $\mathcal{A}$. The reaction from Set 4 always produces E at this step. Thus $W_2 = res_{\mathcal{A}_M}(W_1) = \{E\} \cup \{\text{entity } \rho_i : \text{ reaction } \rho_i \text{ is enabled for } T \text{ in } \mathcal{A}\}$.

For the third step, consider $res_{\mathcal{A}_M}(W_2)$. Only Set 5 is applicable. Since W_2 includes, aside from E, exactly all of the entities ρ_i corresponding to reactions that are not enabled for T in $\mathcal{A}$, the reactions from Set 5 will produce $W_3 = res_{\mathcal{A}} = f_{\mathcal{A}}$.

Since $W_3 \in S$, the sequence continues such that $f_{\mathcal{A}} = f^3_{\mathcal{A}_M}$. □

As we saw in the construction of $\mathcal{A}_M$, the background set S' is much larger than S and the reaction set A' can get big too. The following corollary from [20] gives us a bound for $|S'|$ and $|A'|$.

Corollary 5.5. *Let $\mathcal{A}$ and $\mathcal{A}_M$ be as in Theorem 5.4, and let $\mathcal{A}$ consist of $k \geq 1$ reactions. Then the cardinality of the background set (resp. the reaction set) of $\mathcal{A}_M$ is at most:*

$$2{\cdot}|S| + k + 3 \quad (\textit{resp. } 3{\cdot}|S| + k + 1).$$

Shortly after the publication of [20], the same author published in [21] a two-step simulation of reaction systems by resource-minimal ones. The construction proof is somewhat similar to the one for Theorem 5.4 and can be found in [21].

Theorem 5.6. *For every reaction system $\mathcal{A}$ a resource-minimal reaction system $\mathcal{A}_M$ can be effectively constructed such that whenever $T_0 \Rightarrow_{\mathcal{A}} T_1$, then $T_0 \Rightarrow_{\mathcal{A}_M} U_0 \Rightarrow_{\mathcal{A}_M} T_1$. Moreover, the set U_0 does not contain elements of the background set of $\mathcal{A}$.*

The notation $T_0 \Rightarrow_{\mathcal{A}} T_1$ is equivalent to $T_1 = res_{\mathcal{A}}(T_0) = f_{\mathcal{A}}(T_0)$, and the notation $T_0 \Rightarrow_{\mathcal{A}_M} U_0 \Rightarrow_{\mathcal{A}_M} T_1$ is equivalent to $T_1 = res^2_{\mathcal{A}_M}(T_0) = f^2_{\mathcal{A}_M}(T_0)$. So to compare with Theorem 5.4, whereas that theorem simulates $\mathcal{A}$ with a 3-step process, Theorem 5.6 uses a 2-step process. But still, Theorem 5.6 requires an inflation of the background and reaction sets of $\mathcal{A}_M$, the following corollary tell us what to expect.

Corollary 5.7. *Assume that the background set and the set of reactions of an arbitrary reaction system $\mathcal{A}$ are of cardinalities s and k, respectively. Then a resource-minimal reaction system $\mathcal{A}_M$ satisfying Theorem 5.6 can be effectively constructed such that the cardinalities of its background and reaction sets are $s + k + 1$ and $s^2 + k$, respectively.*

Corollary 5.7 is stated in [21] as following directly from the construction proof of Theorem 5.6. Also mentioned is the implication of this corollary that the feasibility of simulating arbitrary reaction systems by this two-step resource-minimal one allows some of the results on computational complexity of reaction systems to be extended to resource-minimal ones.

5.2.2 Example

Consider the RS $\mathcal{A}$ with the reaction set shown in Table 3.4. This RS does not appear to be resource-minimal. However, using RS Tools and Espresso, we find that the RS is functionally equivalent to a slightly smaller RS:

$a_1 = (\{c\}, \{a, b\}, \{a, b\})$ $\quad a_2 = (\{b\}, \{a\}, \{a\})$ $\quad a_3 = (\{a\}, \{b\}, \{b\}),$

But these reactions are still not resource-minimal. If we need a resource-minimal RS which can produce the same sequence at the expense of getting it from a 3-step process, then we can apply Theorem 5.4, as demonstrated below.

1. Define new entities ρ_1, ρ_2, ρ_3 which are associated with reactions $a_1, a_2,$ and a_3.
2. Define new entities $\overline{a}, \overline{b},$ and $\overline{c}$ and let $\overline{S} = \{\overline{a}, \overline{b}, \overline{c}\}$.
3. Defined new entities $B, \$,$ and E.
4. Let $S' = S \cup S' \cup \{B, \$, E\}$.
5. Include the following reactions in the set A' according to the steps in Theorem 5.4:
   $a_{M_1} = (\{a\}, \{\$\}, \{\overline{a}, B\})$ $\quad a_{M_7} = (\{\overline{a}\}, \{\$\}, \{\rho_1, \rho_2\})$
   $a_{M_2} = (\{b\}, \{\$\}, \{\overline{b}, B\})$ $\quad a_{M_8} = (\{\overline{b}\}, \{\$\}, \{\rho_1, \rho_3\})$
   $a_{M_3} = (\{c\}, \{\$\}, \{\overline{c}, B\})$ $\quad a_{M_9} = (\{B\}, \{\$\}, \{E\})$
   $a_{M_4} = (\{B\}, \{\overline{a}\}, \{\rho_3\})$ $\quad a_{M_{10}} = (\{E\}, \{\rho_1\}, \{a, b\})$
   $a_{M_5} = (\{B\}, \{\overline{b}\}, \{\rho_2\})$ $\quad a_{M_{11}} = (\{E\}, \{\rho_2\}, \{a\})$
   $a_{M_6} = (\{B\}, \{\overline{c}\}, \{\rho_1\})$ $\quad a_{M_{12}} = (\{E\}, \{\rho_3\}, \{b\})$
6. Defined RS $\mathcal{A}' = (S', A')$.

The following table compares $res_{\mathcal{A}}(T)$ with $res^3_{\mathcal{A}'}(T)$ for each $T \in \mathcal{P}(S)$ where all three steps of $res^3_{\mathcal{A}'}$ are shown. For this example, Theorem 5.4 is verified as $res_{\mathcal{A}} = res^3_{\mathcal{A}'}$.

T	$res_{\mathcal{A}}(T)$	$res^1_{\mathcal{A}'}(T)$	$res^2_{\mathcal{A}'}(T)$	$res^3_{\mathcal{A}'}(T)$
$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$	$\varnothing$
$\{a\}$	$\{b\}$	$\{\overline{a}, B\}$	$\{\rho_1, \rho_2, E\}$	$\{b\}$
$\{b\}$	$\{a\}$	$\{\overline{b}, B\}$	$\{\rho_1, \rho_3, E\}$	$\{a\}$
$\{a, b\}$	$\varnothing$	$\{\overline{a}, \overline{b}, B\}$	$\{\rho_1, \rho_2, \rho 3, E\}$	$\varnothing$
$\{c\}$	$\{a, b\}$	$\{\overline{c}, B\}$	$\{\rho_2, \rho_3, E\}$	$\{a, b\}$
$\{a, c\}$	$\{b\}$	$\{\overline{a}, \overline{c}, B\}$	$\{\rho_1, \rho_2, E\}$	$\{b\}$
$\{b, c\}$	$\{a\}$	$\{\overline{b}, \overline{c}, B\}$	$\{\rho_1, \rho_3, E\}$	$\{a\}$
$\{a, b, c\}$	$\varnothing$	$\{\overline{a}, \overline{b}, \overline{c}, B\}$	$\{\rho_1, \rho_2, \rho_3, E\}$	$\varnothing$

6. RS Tools: Exploring reaction systems through software

RS Tools is original software that was written for the purpose of exploring this thesis topic. The process of writing and using the software provided insight into the way reaction systems work. There is much that could be said about the evolution of RS Tools but what is presented here is a description of the tools that are in its toolbox and how they were used in this paper.

6.1 Capabilities of RS Tools

6.1.1 Input and output formats

RS Tools accepts a reaction system defined by its reactions which can be entered in one of three formats. RS Tools discerns the background set from the entered reactions. The reactions can be entered using their conventional notation. They can also be entered using a simpler "plain format" as defined and used by the software *brism* (see [12]), which is a reaction system simulation project which pre-dates RS Tools. By "simulation", we are referring to software simulation of the RS by evaluating the res function and generating sequences with or without a context/interaction sequence. RS Tools also accepts input as a set of Boolean functions in a format similar to that accepted by Logic Friday, which is software that minimizes Boolean logic formulas and generates the corresponding logic circuit diagram.

Once RS Tools parses the given reaction set, a variety of tools are then available. Whether or not the chosen tools produce a new reaction system (e.g. a functionally equivalent minimized RS), the reactions of the resulting RS can be shown in plain format, with conventional notation using plain text, as a set of Boolean functions, or with conventional notation presented in a format that can be copied and pasted into a LaTeX document.

6.1.2 Splitting reactions and "minimizing" Boolean functions"

By splitting reactions, RS Tools generates a functionally equivalent RS with a reaction set each of which produces only one product entity. The proof of Proposition 4.18 describes why and how this can be done. This tool is implied in the use of the tool to "minimize" Boolean functions.

Without using the option to "minimize" Boolean functions, RS Tools parses a reaction

system into a set of Boolean functions such that each one is an indicator for a unique product set among all of the reactions. In other words, for reaction set A, RS Tools builds the set $\{p_i : p_i = P_{a_i} \text{ for every } a_i \in A\}$. RS Tools then generates the Boolean functions each of which is an indicator for one of the subsets p_i.

The intention of the tool to "minimize" Boolean functions is to convert the set of Boolean functions just described into a set of indicator functions for singletons of product entities. In practice, this usually reduces the number of Boolean functions but this tool does not literally minimize them. RS Tools simply splits all of the reactions and re-generates the set of Boolean functions from the split set of reactions.

As an example, the set of unique product sets for the reactions shown in Table 3.5 is $\{\{b, c, d\}, \{b, c\}, \{b\}, \{c, d\}, \{c\}\}$, and there is a Boolean function for each subset (not shown). By "minimizing" the Boolean functions, RS Tools splits all of the reactions and we can see that the unique product sets are then $\{\{b\}, \{c\}, \{d\}\}$ (the entity a does not appear in any of the product sets of the reactions). This results in the generation of three Boolean functions.

6.1.3 Simulating reaction systems

Internally, RS Tools can evaluate the res function for a given subset of the background set for the parsed RS. A context sequence can be entered, again borrowing from the format used by *brism*. If we want to generate the context-independent sequence, we only provide the initial state. But the context sequence input can contain any number of steps.

One of the output options when running a simulation results in a table in LaTeX format. For example, RS Tools was used to produce most of the LaTeX code for Table 3.3.

There are options to simulate the RS before and after an input RS is minimized.

6.1.4 Generating graphs

The 0-context graphs in this paper were generated by the open-source software *Graphviz* via RS Tools. Graphviz accepts specially formatted input describing the graph to be generated and then it generates the image with an arrangement that is determined by Graphviz.

The role of RS Tools is to generate the vertex and edge sets described in Definition 5.1 and from that generate the input needed by Graphviz.

As an example, for the reaction system described in Table 3.5, RS Tools generates and

displays the following set of edges:

$$\{(\varnothing,\varnothing),(\{b\},\{b,c,d\}),(\{c\},\varnothing),(\{d\},\varnothing),(\{a\},\{b,c\}),(\{b,c\},\{c,d\}),(\{b,d\},\{b,c,d\}),$$
$$(\{a,b\},\{b\}),(\{c,d\},\varnothing),(\{a,c\},\{c\}),(\{a,d\},\{b,c\}),(\{b,c,d\},\{c,d\}),(\{a,b,c\},\varnothing),$$
$$(\{a,b,d\},\{b\}),(\{a,c,d\},\{c\}),(\{a,b,c,d\},\varnothing)\}$$

Next, RS Tools generates the following input for Graphviz. After passing this input to Graphiz for processing, RS Tools loads the image file created and displays it. The actual image produced by Graphviz with the below input is shown in Figure 6.1. Since the Graphviz input is displayed in RS Tools, it can be modified to alter the layout of the graph and processed by Graphviz directly.

```
digraph {
node [shape="ellipse"];
layout="dot";
#layout="neato";
rankdir="LR";
#rankdir="TB";
n0 [label="∅"];
n1 [label="{b}"];
n2 [label="{b, c, d}"];
n3 [label="{c}"];
n4 [label="{d}"];
n5 [label="{a}"];
n6 [label="{b, c}"];
n7 [label="{c, d}"];
n8 [label="{b, d}"];
n9 [label="{a, b}"];
n10 [label="{a, c}"];
n11 [label="{a, d}"];
n12 [label="{a, b, c}"];
n13 [label="{a, b, d}"];
n14 [label="{a, c, d}"];
n15 [label="{a, b, c, d}"];
n0 -> n0;
n1 -> n2;
```

```
n3 -> n0;
n4 -> n0;
n5 -> n6;
n6 -> n7;
n8 -> n2;
n9 -> n1;
n7 -> n0;
n10 -> n3;
n11 -> n6;
n2 -> n7;
n12 -> n0;
n13 -> n1;
n14 -> n3;
n15 -> n0;}
```

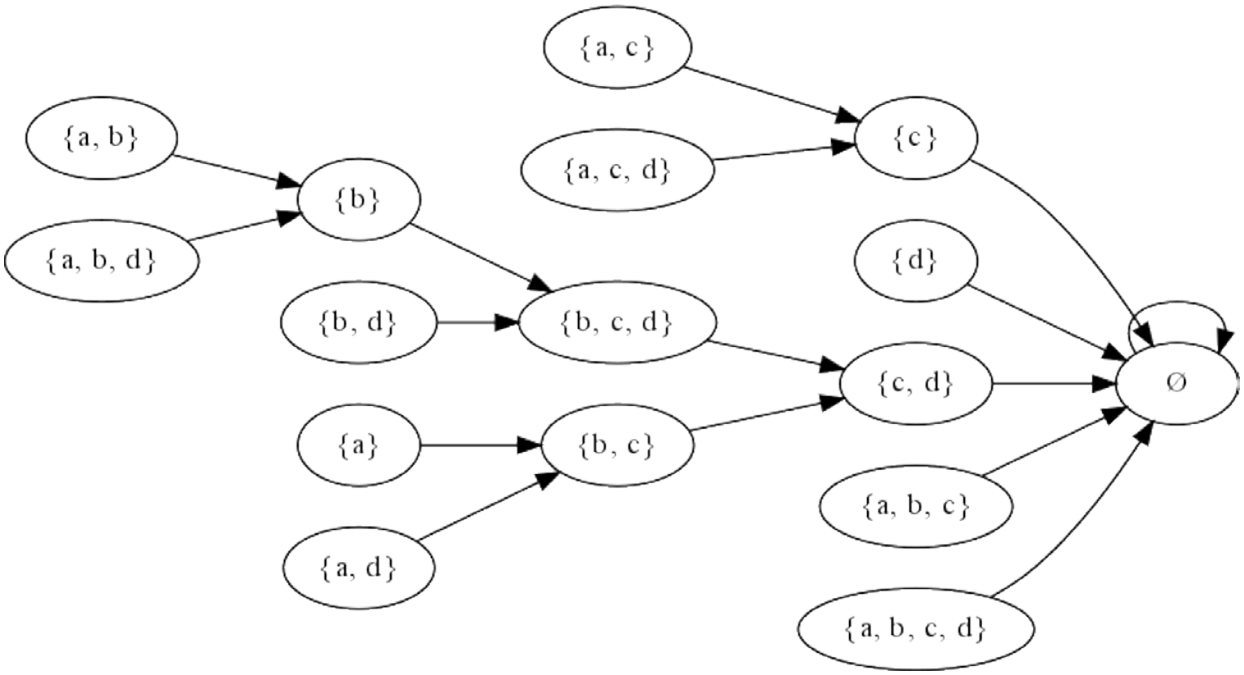

Figure 6.1: Graph generated by Graphviz via RS Tools.

6.1.5 Generating maximally inhibited RS

This tool was very easy to implement since RS Tools already had the ability to generate the vertex and edge sets needed to create a 0-context graph. Once RS Tools has generated those

sets, it simply works through all of the edges and generates a maximally inhibited reaction for each one.

6.1.6 Checking subadditive properties

RS Tools checks for union-subadditivity by verifying that the definition for union-subadditivity holds for every pair of subsets of the background set. It does not need to literally check every pair of subsets. For example, if subsets X and Y are checked, then Y and X do not need to be checked. Also, there is no need to check any pair where one of the subsets is $\varnothing$. For any non-empty subset X, $f(X \cup \varnothing) \subseteq f(X) \cup f(\varnothing)$ always holds. There is also no need to check any pair where $X \subseteq Y$ or $Y \subseteq X$ since they will always pass the test.

The algorithm developed for RS Tools to work through the pairs of subsets that it needs to check for subadditivity does not generate any large arrays of subsets, such as the power set, in memory. In fact, the memory usage, aside from some small overhead, increases by only one bit for each additional increase in cardinality of the background set. To see why this is important, consider a reaction system with a background set S such that $|S| = 30$. Then $|\mathcal{P}(S)| = 2^{30}$. Representing each subset as 30 bits in a 32-bit structure, four gigabytes of memory would be required.

6.1.7 Minimizing reaction systems

Initially, RS Tools was created to convert a reaction system into an equivalent set of Boolean functions. As it turns out, the most important task for RS Tools is to be the front-end visual interface for the back-end logic minimization software. For this thesis, the only logic minimization software used was Espresso.

The input for Espresso is not a set of Boolean functions; the input is a truth table. So an important preparatory task which RS Tools performs is to generate the truth tables needed as input by Espresso. When the input reaction system is parsed by RS Tools, it generates the equivalent set of Boolean functions as already described. Part of this process is determining which subset will be indicated by each Boolean function. For that set of subsets, RS Tools works through that power set and evaluates the Boolean functions, generating a truth table. The truth table has multiple inputs and usually multiple outputs depending on the options used. There is an output column in the truth table for each Boolean function.

Although Espresso has the ability to understand "don't care" states in the input truth table, RS Tools does not currently optimize the truth table to take advantage of this. This would be a useful addition to RS Tools for reaction systems which would generate an extremely large truth table without "don't care" states but would be much smaller if "don't care" states were taken into account.

An option available in RS Tools is to minimize the reaction system by minimizing each Boolean function individually with Espresso. RS Tools prepares for this by generating a truth table for each Boolean function separately. This results in each truth table having one output. RS Tools sends each truth table to Espresso separately.

When Espresso has computed a solution, it returns a truth table. They usually include "don't care" states, and RS Tools recognizes and interprets them. The interpretation of the truth table returned by Espresso involves generating a reaction for each line in the truth table. RS Tools does not generate Boolean functions directly from the output of Espresso, but after RS Tools generates the minimized set of reactions, it can generate (if desired) the equivalent set of Boolean functions in the same way it does when it first parses the input reaction system.

If the option to minimize each Boolean function separately was used, the combined set of reactions generated from the multiple truth tables returned by Espresso can sometimes result in reactions that can be combined. RS Tools checks to see if this can be done and combines reactions where possible in accordance with Proposition 4.19.

The final display of the minimized reactions can be provided in plain format, conventional notation in plain text, or as LaTeX code.

As described in Section 5.1, Espresso has an option that, when enabled, will result in the guaranteed optimally minimized outcome. RS Tools does not currently utilize this "exact" option. If deemed useful for the purposes of studying reaction systems, the option can be added.

6.1.8 Generating RS to produce Finonacci sequence

In an attempt to create an interesting example of a reaction system, a small RS was created to generate the first few Fibonacci numbers, in a multi-step process, that could be represented by a tiny number of binary digits. It was observed that the additional reactions needed to generate more numbers has a pattern that lends itself for automating the generation of the reactions. To accomplish this with RS Tools, the code shown in Appendix A.3 was written and added. An example is also shown in the appendix.

7. Conclusion

The theoretical model for reaction systems has been described as "a new kind of mechanism for generating functions and sequences over a finite set" [15]. Originally introduced as a new way to mathematically model biochemical processes which take place in living cells, the theoretical model has been found to be useful in both abstract and practical ways.

In this thesis, we have presented the formal model for reaction systems and primarily studied its functional behavior. We briefly looked at an application of modeling the reaction that leads to the expression of a gene in E. coli cells which enables the metabolization of lactose.

As sequence-generating functions, we looked at the capabilities of reaction systems to generate potentially long and elaborate sequences. We also looked at sequences generated by interactive reaction systems. We discussed how reaction systems can be viewed as finite state transition systems and represented as 0-context graphs which can also be considered as state transition maps. Concerning functional equivalence, we showed how any reaction system can be represented as a set of Boolean functions. As an aside, we looked at how the representation as Boolean functions leads to representing a reaction system as a logic circuit.

Reaction systems' ability to be represented as Boolean functions eventually lead us to a procedure for minimizing reaction systems using logic minimization algorithms with a focus on one particularly efficient and versatile algorithm called Espresso. The smallest class of reaction systems discussed in this thesis are resource-minimal. In our pursuit of attempting to minimize reaction systems as much as possible, even down to resource-minimality if possible, we discussed the main result which is the focus of this thesis: a characterization theorem, Theorem 4.17 which was originally proved in [5], to determine if there exists a resource-minimal reaction system which is functionally equivalent to the one we want to minimize.

Original software called RS Tools was created for the purpose of exploring this thesis topic. We discussed some of the ways RS Tools was used to provide insight and even create some of the content for this thesis.

A. Appendix

A.1 An in-depth look at minimizing a RS

In Section 5.1.1, the minimization of the reaction system shown in Table 5.3, preceded by a verification of its subadditive properties, is discussed. This appendix shows the details of what RS Tools tells us about the RS before it is minimized, what RS Tools does in preparation for minimizing the RS using Espresso, the input to and output from Espresso, and how RS Tools interprets the output.

The algorithm used by RS Tools to check for subadditivity skips checking some pairs of subsets of the background set which can greatly reduce the number of pairs it needs to check for subadditivity. For example, if it verifies $f(X \cup Y) \subseteq f(X) \cup f(Y)$, then it does not need to check $f(Y \cup X)$. For the reaction set we are using, 1024 pairs of subsets was reduced to 435.

The first step RS Tools performs after parsing the input reaction set is to convert the input RS into a set of Boolean functions. It determines the set of unique product sets and generates a Boolean function for each. In this case, RS Tools reports the following.

```
The boolean functions are:
f1 = (e4∧e5∧¬e1∧¬e2∧¬e3) ∨ (e1∧e4∧e5∧¬e2∧¬e3)
f2 = (e2∧e4∧e5∧¬e1∧¬e3) ∨ (e1∧e2∧e4∧e5∧¬e3)
f3 = (e3∧e5∧¬e1∧¬e2∧¬e4) ∨ (e3∧e4∧¬e5∧¬e1∧¬e2) ∨ (e3∧e4∧e5∧¬e1∧¬e2) ∨
 (e1∧e3∧e4∧¬e5∧¬e2) ∨ (e1∧e3∧e4∧e5∧¬e2) ∨ (e1∧e2∧e3∧e4∧¬e5)
f4 = (e5∧¬e1∧¬e2∧¬e4∧¬e3) ∨ (e1∧e5∧¬e2∧¬e4∧¬e3) ∨ (e2∧e5∧¬e1∧¬e4∧¬e3) ∨
 (e1∧e2∧e5∧¬e4∧¬e3)
f5 = (e3∧¬e5∧¬e1∧¬e2∧¬e4) ∨ (e2∧e3∧¬e5∧¬e1∧¬e4) ∨ (e2∧e3∧e5∧¬e1∧¬e4) ∨
 (e2∧e3∧e4∧¬e5∧¬e1) ∨ (e2∧e3∧e4∧e5∧¬e1)
f6 = (e1∧e2∧¬e5∧¬e4∧¬e3) ∨ (e1∧e2∧e3∧¬e5∧¬e4)
f7 = (e1∧¬e5∧¬e2∧¬e4∧¬e3) ∨ (e1∧e3∧¬e5∧¬e2∧¬e4) ∨ (e1∧e3∧e5∧¬e2∧¬e4)
f8 = (e1∧e2∧e3∧e5∧¬e4)
f9 = (e4∧¬e5∧¬e1∧¬e2∧¬e3) ∨ (e1∧e4∧¬e5∧¬e2∧¬e3) ∨ (e2∧e4∧¬e5∧¬e1∧¬e3) ∨
 (e1∧e2∧e4∧¬e5∧¬e3)
f10 = (e2∧¬e5∧¬e1∧¬e4∧¬e3)
where the product sets associated with each fn are
f1: {e1, e2, e3, e4, e5}
```

```
f2: {e1, e2, e3, e4}
f3: {e1, e3, e4, e5}
f4: {e1, e3, e4}
f5: {e1, e3, e5}
f6: {e1, e4, e5}
f7: {e1, e4}
f8: {e1}
f9: {e2, e3, e4, e5}
f10: {e5}
```

There are some options available which may affect the final solution. We can choose to have Espresso minimize the Boolean functions separately instead together, we can choose to minimize the number of Boolean functions before sending any input to Espresso, or both. Without using either option, RS Tools reports, after minimizing the RS with Espresso, the following reaction set was reported. The solution from Espresso resulted in an RS with minimized number of reactions, but it a $(2, 1)$ rs.

```
a1=({e2}, {e5}, {e5})
a2=({e1}, {e4}, {e1})
a3=({e3, e4}, {e5}, {e1, e3})
a4=({e1}, {e5}, {e4})
a5=({e5}, {e2}, {e1, e4})
a6=({e4}, {e3}, {e2, e3, e4})
a7=({e3}, {e1}, {e1, e3, e5})
a8=({e4}, {e2}, {e3, e4, e5})
a9=({e5}, {e3}, {e1, e3, e4})
```

If the option to minimize the Boolean functions separately is enabled without enabling the option to minimize the number of Boolean functions, RS reports the following as the solution. As of this writing, whether RS Tools is properly using Espresso for this option has not been investigated, but the RS produced from Espresso's solution does not appear to have any advantages.

```
a1=({e4, e5}, {e2, e3}, {e1, e2, e3, e4, e5})
a2=({e2, e4, e5}, {e3}, {e1, e2, e3, e4})
a3=({e3, e5}, {e1, e2}, {e1, e3, e4, e5})
a4=({e3, e1, e4}, {e5}, {e1, e3, e4, e5})
a5=({e3, e4}, {e2}, {e1, e3, e4, e5})
```

```
a6=({e5}, {e4, e3}, {e1, e3, e4})
a7=({e3}, {e5, e1, e4}, {e1, e3, e5})
a8=({e3, e2}, {e1}, {e1, e3, e5})
a9=({e1, e2}, {e5, e4}, {e1, e4, e5})
a10=({e1}, {e5, e2, e4}, {e1, e4})
a11=({e1, e3}, {e2, e4}, {e1, e4})
a12=({e1, e2, e3, e5}, {e4}, {e1})
a13=({e4}, {e5, e3}, {e2, e3, e4, e5})
a14=({e2}, {e5, e1, e4, e3}, {e5})
```

Using both options results in a resource-minimal rs, which we expect since the maximally inhibited RS is union- and intersection-subadditive. So we will discuss what happens when both options are enabled. The option to minimize Boolean functions enables RS Tools to perform the following. First it splits all of the input reactions so that each one produces one product entity. Then it generates a set of Boolean functions based on the expanded rs. For our example, RS Tools reports the following, which has been edited to not show all 99 split reactions, but the corresponding Boolean functions are shown in full.

```
To minimize boolean functions, some of the reactions were split.
The equivalent reaction system is:
e5,e1 e2 e4 e3,e3
e1,e5 e2 e4 e3,e1
e2,e5 e1 e4 e3,e5
e4,e5 e1 e2 e3,e2
e3,e5 e1 e2 e4,e5
.
.
.
e2 e4,e5 e1 e3,e2
e1 e4 e5,e2 e3,e2
e2 e4 e5,e1 e3,e2
e1 e2 e4,e5 e3,e2
e1 e2 e4 e5,e3,e2

The boolean funtions are:
f1 = (e1∧¬e5∧¬e2∧¬e4∧¬e3) ∨ (e1∧e2∧e3∧e5∧¬e4) ∨ (e3∧¬e5∧¬e1∧¬e2∧¬e4) ∨
 (e1∧e5∧¬e2∧¬e4∧¬e3) ∨ (e2∧e5∧¬e1∧¬e4∧¬e3) ∨ (e4∧e5∧¬e1∧¬e2∧¬e3) ∨
```

```
 (e3∧e5∧¬e1∧¬e2∧¬e4) ∨ (e1∧e2∧¬e5∧¬e4∧¬e3) ∨ (e1∧e3∧¬e5∧¬e2∧¬e4) ∨
 (e2∧e3∧¬e5∧¬e1∧¬e4) ∨ (e3∧e4∧¬e5∧¬e1∧¬e2) ∨ (e1∧e2∧e5∧¬e4∧¬e3) ∨
 (e1∧e4∧e5∧¬e2∧¬e3) ∨ (e1∧e3∧e5∧¬e2∧¬e4) ∨ (e2∧e4∧e5∧¬e1∧¬e3) ∨
 (e2∧e3∧e5∧¬e1∧¬e4) ∨ (e3∧e4∧e5∧¬e1∧¬e2) ∨ (e1∧e2∧e3∧¬e5∧¬e4) ∨
 (e1∧e3∧e4∧¬e5∧¬e2) ∨ (e2∧e3∧e4∧¬e5∧¬e1) ∨ (e1∧e2∧e4∧e5∧¬e3) ∨
 (e1∧e3∧e4∧e5∧¬e2) ∨ (e2∧e3∧e4∧e5∧¬e1) ∨ (e1∧e2∧e3∧e4∧¬e5) ∨
 (e5∧¬e1∧¬e2∧¬e4∧¬e3)
f2 = (e4∧¬e5∧¬e1∧¬e2∧¬e3) ∨ (e4∧e5∧¬e1∧¬e2∧¬e3) ∨ (e1∧e4∧¬e5∧¬e2∧¬e3) ∨
 (e2∧e4∧¬e5∧¬e1∧¬e3) ∨ (e1∧e4∧e5∧¬e2∧¬e3) ∨ (e2∧e4∧e5∧¬e1∧¬e3) ∨
 (e1∧e2∧e4∧¬e5∧¬e3) ∨ (e1∧e2∧e4∧e5∧¬e3)
f3 = (e5∧¬e1∧¬e2∧¬e4∧¬e3) ∨ (e4∧¬e5∧¬e1∧¬e2∧¬e3) ∨ (e3∧¬e5∧¬e1∧¬e2∧¬e4) ∨
 (e1∧e5∧¬e2∧¬e4∧¬e3) ∨ (e2∧e5∧¬e1∧¬e4∧¬e3) ∨ (e4∧e5∧¬e1∧¬e2∧¬e3) ∨
 (e3∧e5∧¬e1∧¬e2∧¬e4) ∨ (e1∧e4∧¬e5∧¬e2∧¬e3) ∨ (e2∧e4∧¬e5∧¬e1∧¬e3) ∨
 (e2∧e3∧¬e5∧¬e1∧¬e4) ∨ (e3∧e4∧¬e5∧¬e1∧¬e2) ∨ (e1∧e2∧e5∧¬e4∧¬e3) ∨
 (e1∧e4∧e5∧¬e2∧¬e3) ∨ (e2∧e4∧e5∧¬e1∧¬e3) ∨ (e2∧e3∧e5∧¬e1∧¬e4) ∨
 (e3∧e4∧e5∧¬e1∧¬e2) ∨ (e1∧e2∧e4∧¬e5∧¬e3) ∨ (e1∧e3∧e4∧¬e5∧¬e2) ∨
 (e2∧e3∧e4∧¬e5∧¬e1) ∨ (e1∧e2∧e4∧e5∧¬e3) ∨ (e1∧e3∧e4∧e5∧¬e2) ∨
 (e2∧e3∧e4∧e5∧¬e1) ∨ (e1∧e2∧e3∧e4∧¬e5)
f4 = (e1∧e5∧¬e2∧¬e4∧¬e3) ∨ (e2∧e5∧¬e1∧¬e4∧¬e3) ∨ (e4∧e5∧¬e1∧¬e2∧¬e3) ∨
 (e3∧e5∧¬e1∧¬e2∧¬e4) ∨ (e1∧e2∧¬e5∧¬e4∧¬e3) ∨ (e1∧e4∧¬e5∧¬e2∧¬e3) ∨
 (e1∧e3∧¬e5∧¬e2∧¬e4) ∨ (e2∧e4∧¬e5∧¬e1∧¬e3) ∨ (e3∧e4∧¬e5∧¬e1∧¬e2) ∨
 (e1∧e2∧e5∧¬e4∧¬e3) ∨ (e1∧e4∧e5∧¬e2∧¬e3) ∨ (e1∧e3∧e5∧¬e2∧¬e4) ∨
 (e2∧e4∧e5∧¬e1∧¬e3) ∨ (e3∧e4∧e5∧¬e1∧¬e2) ∨ (e1∧e2∧e4∧¬e5∧¬e3) ∨
 (e1∧e2∧e3∧¬e5∧¬e4) ∨ (e1∧e3∧e4∧¬e5∧¬e2) ∨ (e1∧e2∧e4∧e5∧¬e3) ∨
 (e1∧e3∧e4∧e5∧¬e2) ∨ (e1∧e2∧e3∧e4∧¬e5) ∨ (e5∧¬e1∧¬e2∧¬e4∧¬e3) ∨
 (e1∧¬e5∧¬e2∧¬e4∧¬e3) ∨ (e4∧¬e5∧¬e1∧¬e2∧¬e3)
f5 = (e2∧¬e5∧¬e1∧¬e4∧¬e3) ∨ (e3∧¬e5∧¬e1∧¬e2∧¬e4) ∨ (e2∧e3∧¬e5∧¬e1∧¬e4) ∨
 (e2∧e3∧e5∧¬e1∧¬e4) ∨ (e2∧e3∧e4∧¬e5∧¬e1) ∨ (e2∧e3∧e4∧e5∧¬e1) ∨
 (e4∧¬e5∧¬e1∧¬e2∧¬e3) ∨ (e4∧e5∧¬e1∧¬e2∧¬e3) ∨ (e3∧e5∧¬e1∧¬e2∧¬e4) ∨
 (e1∧e2∧¬e5∧¬e4∧¬e3) ∨ (e1∧e4∧¬e5∧¬e2∧¬e3) ∨ (e2∧e4∧¬e5∧¬e1∧¬e3) ∨
 (e3∧e4∧¬e5∧¬e1∧¬e2) ∨ (e1∧e4∧e5∧¬e2∧¬e3) ∨ (e3∧e4∧e5∧¬e1∧¬e2) ∨
 (e1∧e2∧e4∧¬e5∧¬e3) ∨ (e1∧e2∧e3∧¬e5∧¬e4) ∨ (e1∧e3∧e4∧¬e5∧¬e2) ∨
 (e1∧e3∧e4∧e5∧¬e2) ∨ (e1∧e2∧e3∧e4∧¬e5)
where the product sets associated with each fn are
f1: {e1}
f2: {e2}
```

```
f3: {e3}
f4: {e4}
f5: {e5}
```

Next, RS Tools calls Espresso for each of the five Boolean functions. Following is the actual input file for f1 and the output from Espresso. The input file specified the number of input variables, which is $|S| = 5$, and the number of output variables which, in this case, is just one for the function f1. Not using the options we are using would result in multiple outputs but that did not produce a $(1, 1)$ solution as we saw.

```
Minimizer tool (espresso) input file for f1:
.i 5
.o 1
00001 1
00011 1
00101 1
00111 1
01000 1
01001 1
01010 1
01011 1
01100 1
01101 1
01110 1
01111 1
10000 1
10001 1
10011 1
10100 1
10101 1
10111 1
11000 1
11001 1
11010 1
11011 1
11100 1
11101 1
```

```
11110 1
.e


Minimizer tool response for f1 (exit code 0)
.i 5
.o 1
.p 5
0---1 1
--0-1 1
1--0- 1
-0--1 1
-1--0 1
.e
```

The output from Espresso for f1 is interpreted by RS Tools and the following Boolean functions are generated from this output.

```
The boolean functions are now:
f1 = (e3∧¬e2) ∨ (e1∧¬e4) ∨ (e3∧¬e5) ∨ (e3∧¬e1) ∨ (e5∧¬e3)
f2 = (e4∧¬e3)
f3 = (e4∧¬e5) ∨ (e4∧¬e2) ∨ (e3∧¬e1) ∨ (e5∧¬e3)
f4 = (e5∧¬e3) ∨ (e1∧¬e5) ∨ (e5∧¬e2) ∨ (e4∧¬e3) ∨ (e4∧¬e2)
f5 = (e3∧¬e1) ∨ (e2∧¬e5) ∨ (e4∧¬e2)
where the product sets associated with each fn are
f1: {e1}
f2: {e2}
f3: {e3}
f4: {e4}
f5: {e5}
```

RS Tools repeats the process for the other four Boolean functions. Next, RS Tools generates the reaction system corresponding to these Boolean functions. The reaction set is shown below, however some of the reactions can be combined and when RS Tools does this, the final resource-minimal RS is produced as shown in Table 5.4.

```
The minimal reaction system is
a1=({e3}, {e1}, {e5})
```

```
a2=({e3}, {e2}, {e1})
a3=({e1}, {e4}, {e1})
a4=({e3}, {e5}, {e1})
a5=({e5}, {e3}, {e4})
a6=({e4}, {e3}, {e2})
a7=({e4}, {e5}, {e3})
a8=({e4}, {e2}, {e3})
a9=({e1}, {e5}, {e4})
a10=({e5}, {e2}, {e4})
a11=({e2}, {e5}, {e5})
a12=({e3}, {e1}, {e1})
a13=({e3}, {e1}, {e3})
a14=({e5}, {e3}, {e3})
a15=({e4}, {e2}, {e5})
a16=({e5}, {e3}, {e1})
a17=({e4}, {e3}, {e4})
a18=({e4}, {e2}, {e4})
```

A.2 Minimizing a maximally inhibited RS

For the state transition sequence shown in Figure A.1, a maximally inhibited RS was constructed, using the technique in the proof of Theorem 3.4, to generate this cycle. The background set is $\{a, b, c, d, e\}$. The maximally inhibited RS reaction set is shown as Reaction Set A. As was mentioned in the discussion of the theorem, the maximally inhibited RS can seem bloated. However, using RS Tools and Espresso, this RS was simplified from a $(4, 4)$ RS to a $(3, 3)$. Both Reaction Sets B and C are $(3, 3)$ sets, and as you can see, depending on the minimization options used in RS Tools, we either get a functionally equivalent set with fewer reactions (Reaction Set B) or one with more reactions but, as can be visually discerned, has simpler reactions, although it is still $(3, 3)$.

React Set A:

$(\{a,b\},\{c,d,e\},\{b,c,e\})$
$(\{a,b,c,d\},\{e\},\{a,e\})$
$(\{b,c,e\},\{a,d\},\{c\})$
$(\{a,e\},\{b,c,d\},\{b,e\})$
$(\{c\},\{a,b,d,e\},\{a,c\})$
$(\{b,e\},\{a,c,d\},\{e\})$
$(\{a,c\},\{b,d,e\},\{d\})$
$(\{e\},\{a,b,c,d\},\{a,b,e\})$
$(\{d\},\{a,b,c,e\},\{b,c,d,e\})$
$(\{a,b,e\},\{c,d\},\{c,e\})$
$(\{b,c,d,e\},\{a\},\{a\})$
$(\{c,e\},\{a,b,d\},\{a,b,c\})$
$(\{a\},\{b,c,d,e\},\{a,d\})$
$(\{a,b,c\},\{d,e\},\{a,c,e\})$
$(\{a,d\},\{b,c,e\},\{a,b,d,e\})$
$(\{a,c,e\},\{b,d\},\{a,b,c,e\})$
$(\{a,b,d,e\},\{c\},\{b,d\})$
$(\{a,b,c,e\},\{d\},\{b,c\})$
$(\{b,d\},\{a,c,e\},\{b\})$
$(\{b,c\},\{a,d,e\},\{a,d,e\})$
$(\{b\},\{a,c,d,e\},\{a,b,d\})$
$(\{a,d,e\},\{b,c\},\{b,d,e\})$
$(\{a,b,d\},\{c,e\},\{c,d\})$
$(\{b,d,e\},\{a,c\},\{c,d,e\})$
$(\{c,d\},\{a,b,e\},\{a,c,d\})$
$(\{c,d,e\},\{a,b\},\{d,e\})$
$(\{a,c,d\},\{b,e\},\{b,c,d\})$
$(\{d,e\},\{a,b,c\},\{a,c,d,e\})$
$(\{b,c,d\},\{a,e\},\{a,b,c,d\})$
$(\{a,c,d,e\},\{b\},\{a,b\})$

React Set B:

$(\{a,b,d\},\{c,e\},\{d,c\})$
$(\{a,b,c\},\{e\},\{e,a\})$
$(\{a,c,d\},\{b,e\},\{b,c\})$
$(\{c\},\{a,b,e\},\{c,a\})$
$(\{b,c,d\},\{a\},\{a\})$
$(\{e\},\{a,b,d\},\{a,b\})$
$(\{b,c\},\{d,e\},\{e\})$
$(\{a,c,e\},\{b\},\{a,b\})$
$(\{b,c,d\},\{a,e\},\{c,d,b\})$
$(\{a,d,e\},\{c\},\{b,d\})$
$(\{b\},\{c,d,e\},\{b\})$
$(\{d\},\{a,b,c\},\{c,e\})$
$(\{a,c,e\},\{d\},\{b\})$
$(\{a,e\},\{b,d\},\{e,b\})$
$(\{a,b\},\{c,d\},\{e\})$
$(\{c,e\},\{d\},\{c\})$
$(\{e\},\{a,b,c\},\{a\})$
$(\{a,d\},\{b,c\},\{e,b\})$
$(\{d,e\},\{a,b\},\{e\})$
$(\{a,b\},\{d\},\{c\})$
$(\{a\},\{b,c,e\},\{a\})$
$(\{e\},\{c,d\},\{e\})$
$(\{d\},\{a,c,e\},\{b\})$
$(\{d,e\},\{a,c\},\{c,d,e\})$
$(\{b\},\{a,d,e\},\{a,d\})$
$(\{d\},\{a,b\},\{d\})$

React Set C:

$(\{e\},\{b,d,a\},\{a\})$
$(\{b,a\},\{c,e\},\{c\})$
$(\{a\},\{b,c,e\},\{a\})$
$(\{d,e\},\{c,a\},\{c\})$
$(\{e\},\{b,c,a\},\{a\})$
$(\{b,a\},\{d\},\{c\})$
$(\{b\},\{d,a,e\},\{d,a\})$
$(\{c,d\},\{b,e\},\{c\})$
$(\{b,c\},\{e\},\{a\})$
$(\{c,e\},\{d\},\{c\})$
$(\{b,c,d\},\{a\},\{a\})$
$(\{c,d\},\{a,e\},\{c,d\})$
$(\{c,a,e\},\{b\},\{a\})$
$(\{a\},\{b,e\},\{d\})$
$(\{c\},\{a,e\},\{a\})$
$(\{d,e\},\{c\},\{d\})$
$(\{d\},\{c,b,e\},\{b\})$
$(\{d\},\{a,b\},\{d\})$
$(\{d,b\},\{a,e\},\{b\})$
$(\{a,d\},\{c\},\{d\})$
$(\{b\},\{c,d,e\},\{b\})$
$(\{b,a\},\{c,d\},\{e\})$
$(\{e\},\{d,b\},\{b\})$
$(\{b,c\},\{d,e\},\{e\})$
$(\{a,c,e\},\{d\},\{b\})$
$(\{b,c,a\},\{e\},\{e\})$
$(\{a,d\},\{b\},\{b\})$
$(\{d,e\},\{b,a\},\{e\})$
$(\{a,d,e\},\{c\},\{b\})$
$(\{d\},\{b,c\},\{e\})$
$(\{c\},\{b,a,e\},\{c\})$
$(\{e,a\},\{b,d\},\{e\})$
$(\{d\},\{b,a,e\},\{c\})$
$(\{e\},\{c,a\},\{e\})$

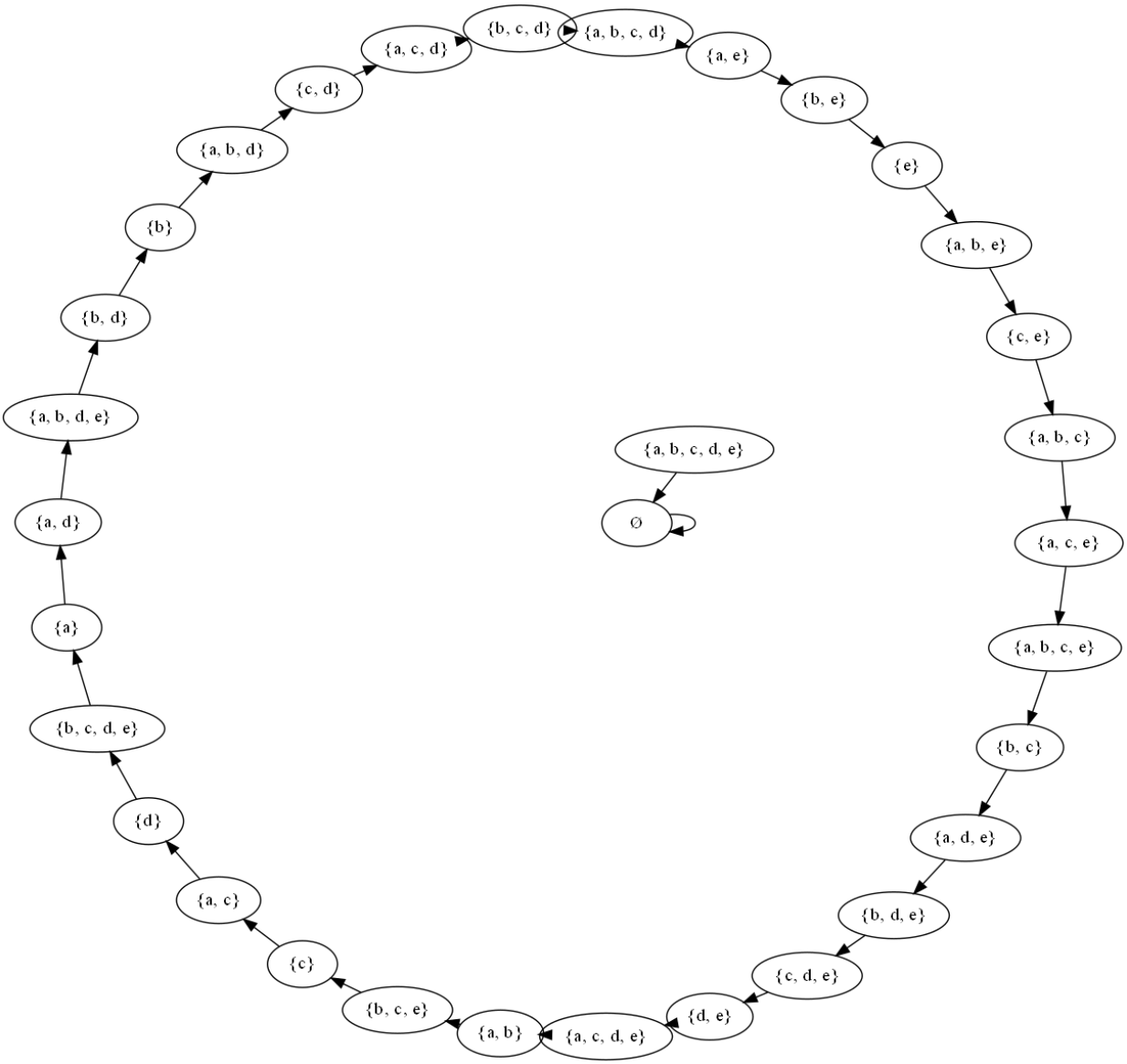

Figure A.1: State transition map for a cycle.

A.3 Fibonacci sequence generated by an RS

This appendix describes a reaction system, the reactions for which can be generated by RS Tools, which generates a sequence of Fibonacci numbers.

Proposition. *For an integer $N \geq 2$, there exists a reaction system $\mathcal{A}$ which generates a sequence of Fibonacci numbers, $\{F_i\}, i = 3, 4, ...$, such that $F_i < 2^N$, in an (N+1)-step process with initial state $T = \{a0, b0, y0\}$. Define $F_0 = 0$, $F_1 = 1$, and $F_2 = 1$, then F_i can be obtained from $res_{\mathcal{A}}^{(i-2)*(N+1)-1}, i = 3, 4, ...$*

This proposition has not yet been formally proved, although the algorithm currently used by RS Tools to generate the RS for any $N \geq 2$ is shown below and can be demonstrated to generate an RS that holds for the claim.

```
procedure CreateFibonacciRS(N: integer);
var
  i: integer;
  s: string;
  s2: string;
  Latex: boolean;
begin

  with RSOutput.Lines do
  begin
    for i := 0 to N-1 do
      Add('a' + inttostr(i) + ', Z X, a' + inttostr(i));
    for i := 0 to N-1 do
      Add('b' + inttostr(i) + ', Z X, b' + inttostr(i));
    for i := 0 to N-1 do
      Add('d' + inttostr(i) + ', Z X, d' + inttostr(i));
    Add('y0 a0, b0 X, d0 y1');
    Add('y0 b0, a0 X, d0 y1');
    Add('y0 a0 b0, X, c1 y1');
    Add('y0, a0 b0 X, y1');

    for i := 1 to N-2 do
    begin
```

```
    s := IntToStr(i);
    s2 := IntToStr(i+1);
    Add(format('y%s a%s, b%s c%s, d%s y%s', [s, s, s, s, s, s2])
        );
    Add(format('y%s b%s, a%s c%s, d%s y%s', [s, s, s, s, s, s2])
        );
    Add(format('y%s a%s b%s, c%s, c%s y%s', [s, s, s, s, s2, s2
        ]));
    Add(format('y%s a%s c%s, b%s, c%s y%s', [s, s, s, s, s2, s2
        ]));
    Add(format('y%s b%s c%s, a%s, c%s y%s', [s, s, s, s, s2, s2
        ]));
    Add(format('y%s a%s b%s c%s, y%s, d%s c%s y%s', [s, s, s, s,
         IntToStr(i-1), s, s2, s2]));
    Add(format('y%s c%s, a%s b%s, d%s y%s', [s, s, s, s, s, s2])
        );
    Add(format('y%s, a%s b%s c%s, y%s', [s, s, s, s, s2]));
end;

s := IntToStr(N-1);
s2 := IntToStr(N);
Add(format('y%s a%s, b%s c%s, d%s Z', [s, s, s, s, s]));
Add(format('y%s b%s, a%s c%s, d%s Z', [s, s, s, s, s]));
Add(format('y%s a%s b%s, c%s, d%s X', [s, s, s, s, s2]));
Add(format('y%s a%s c%s, b%s, d%s X', [s, s, s, s, s2]));
Add(format('y%s b%s c%s, a%s, d%s X', [s, s, s, s, s2]));
Add(format('y%s a%s b%s c%s, y%s, d%s d%s X', [s, s, s, s,
   IntToStr(i-1), s, s2]));
Add(format('y%s c%s, a%s b%s, d%s Z', [s, s, s, s, s]));
Add(format('y%s, a%s b%s c%s, Z', [s, s, s, s]));

for i := 0 to N-1 do
  Add('Z b' + inttostr(i) + ', X, a' + inttostr(i));
for i := 0 to N-1 do
  Add('Z d' + inttostr(i) + ', X, b' + inttostr(i));
Add('Z, X, y0');
```

```
    end;
    ContextSequence.Lines.Add('a0 b0 y0');
end;
```

The reactions generated for N=5 are shown below.

$a_1 = (\{a0\}, \{Z, X\}, \{a0\})$
$a_2 = (\{a1\}, \{Z, X\}, \{a1\})$
$a_3 = (\{a2\}, \{Z, X\}, \{a2\})$
$a_4 = (\{a3\}, \{Z, X\}, \{a3\})$
$a_5 = (\{a4\}, \{Z, X\}, \{a4\})$
$a_6 = (\{b0\}, \{Z, X\}, \{b0\})$
$a_7 = (\{b1\}, \{Z, X\}, \{b1\})$
$a_8 = (\{b2\}, \{Z, X\}, \{b2\})$
$a_9 = (\{b3\}, \{Z, X\}, \{b3\})$
$a_{10} = (\{b4\}, \{Z, X\}, \{b4\})$
$a_{11} = (\{d0\}, \{Z, X\}, \{d0\})$
$a_{12} = (\{d1\}, \{Z, X\}, \{d1\})$
$a_{13} = (\{d2\}, \{Z, X\}, \{d2\})$
$a_{14} = (\{d3\}, \{Z, X\}, \{d3\})$
$a_{15} = (\{d4\}, \{Z, X\}, \{d4\})$
$a_{16} = (\{y0, a0\}, \{b0, X\}, \{d0, y1\})$
$a_{17} = (\{y0, b0\}, \{a0, X\}, \{d0, y1\})$
$a_{18} = (\{y0, a0, b0\}, \{X\}, \{c1, y1\})$
$a_{19} = (\{y0\}, \{a0, b0, X\}, \{y1\})$
$a_{20} = (\{y1, a1\}, \{b1, c1\}, \{d1, y2\})$
$a_{21} = (\{y1, b1\}, \{a1, c1\}, \{d1, y2\})$
$a_{22} = (\{y1, a1, b1\}, \{c1\}, \{c2, y2\})$
$a_{23} = (\{y1, a1, c1\}, \{b1\}, \{c2, y2\})$
$a_{24} = (\{y1, b1, c1\}, \{a1\}, \{c2, y2\})$
$a_{25} = (\{y1, a1, b1, c1\}, \{y0\}, \{d1, c2, y2\})$
$a_{26} = (\{y1, c1\}, \{a1, b1\}, \{d1, y2\})$
$a_{27} = (\{y1\}, \{a1, b1, c1\}, \{y2\})$
$a_{28} = (\{y2, a2\}, \{b2, c2\}, \{d2, y3\})$
$a_{29} = (\{y2, b2\}, \{a2, c2\}, \{d2, y3\})$
$a_{30} = (\{y2, a2, b2\}, \{c2\}, \{c3, y3\})$
$a_{31} = (\{y2, a2, c2\}, \{b2\}, \{c3, y3\})$
$a_{32} = (\{y2, b2, c2\}, \{a2\}, \{c3, y3\})$
$a_{33} = (\{y2, a2, b2, c2\}, \{y1\}, \{d2, c3, y3\})$
$a_{34} = (\{y2, c2\}, \{a2, b2\}, \{d2, y3\})$
$a_{35} = (\{y2\}, \{a2, b2, c2\}, \{y3\})$
$a_{36} = (\{y3, a3\}, \{b3, c3\}, \{d3, y4\})$
$a_{37} = (\{y3, b3\}, \{a3, c3\}, \{d3, y4\})$
$a_{38} = (\{y3, a3, b3\}, \{c3\}, \{c4, y4\})$
$a_{39} = (\{y3, a3, c3\}, \{b3\}, \{c4, y4\})$
$a_{40} = (\{y3, b3, c3\}, \{a3\}, \{c4, y4\})$
$a_{41} = (\{y3, a3, b3, c3\}, \{y2\}, \{d3, c4, y4\})$
$a_{42} = (\{y3, c3\}, \{a3, b3\}, \{d3, y4\})$
$a_{43} = (\{y3\}, \{a3, b3, c3\}, \{y4\})$
$a_{44} = (\{y4, a4\}, \{b4, c4\}, \{d4, Z\})$
$a_{45} = (\{y4, b4\}, \{a4, c4\}, \{d4, Z\})$
$a_{46} = (\{y4, a4, b4\}, \{c4\}, \{d5, X\})$
$a_{47} = (\{y4, a4, c4\}, \{b4\}, \{d5, X\})$
$a_{48} = (\{y4, b4, c4\}, \{a4\}, \{d5, X\})$
$a_{49} = (\{y4, a4, b4, c4\}, \{y3\}, \{d4, d5, X\})$
$a_{50} = (\{y4, c4\}, \{a4, b4\}, \{d4, Z\})$
$a_{51} = (\{y4\}, \{a4, b4, c4\}, \{Z\})$
$a_{52} = (\{Z, b0\}, \{X\}, \{a0\})$
$a_{53} = (\{Z, b1\}, \{X\}, \{a1\})$
$a_{54} = (\{Z, b2\}, \{X\}, \{a2\})$
$a_{55} = (\{Z, b3\}, \{X\}, \{a3\})$
$a_{56} = (\{Z, b4\}, \{X\}, \{a4\})$
$a_{57} = (\{Z, d0\}, \{X\}, \{b0\})$
$a_{58} = (\{Z, d1\}, \{X\}, \{b1\})$
$a_{59} = (\{Z, d2\}, \{X\}, \{b2\})$
$a_{60} = (\{Z, d3\}, \{X\}, \{b3\})$
$a_{61} = (\{Z, d4\}, \{X\}, \{b4\})$
$a_{62} = (\{Z\}, \{X\}, \{y0\})$

The sequence shown below is for an RS that generates Fibonacci numbers up to 2^5.

To obtain F_i from $res_{\mathcal{A}}^{(i-2)*(N+1)-1}, i = 3, 4, ...$, which is indicated by including the entity Z in the result starting with step 10, the value is obtained from the entities dj (i.e., $d0, d1$, etc) which represent binary digits. For example, the subset $\{d1, d0\}$ represents the binary number 11.

0	$\{a0, b0, y0\}$	$\{a0, b0, y0\}$	$\{a_1, a_6, a_{18}\}$	$\{a0, b0, c1, y1\}$
1	$\{\varnothing\}$	$\{a0, b0, c1, y1\}$	$\{a_1, a_6, a_{26}\}$	$\{a0, b0, d1, y2\}$
2	$\{\varnothing\}$	$\{a0, b0, d1, y2\}$	$\{a_1, a_6, a_{12}, a_{35}\}$	$\{a0, b0, d1, y3\}$
3	$\{\varnothing\}$	$\{a0, b0, d1, y3\}$	$\{a_1, a_6, a_{12}, a_{43}\}$	$\{a0, b0, d1, y4\}$
4	$\{\varnothing\}$	$\{a0, b0, d1, y4\}$	$\{a_1, a_6, a_{12}, a_{51}\}$	$\{a0, b0, d1, Z\}$
5	$\{\varnothing\}$	$\{a0, b0, d1, Z\}$	$\{a_{52}, a_{58}, a_{62}\}$	$\{a0, b1, y0\}$
6	$\{\varnothing\}$	$\{a0, b1, y0\}$	$\{a_1, a_7, a_{16}\}$	$\{a0, b1, d0, y1\}$
7	$\{\varnothing\}$	$\{a0, b1, d0, y1\}$	$\{a_1, a_7, a_{11}, a_{21}\}$	$\{a0, b1, d0, d1, y2\}$
8	$\{\varnothing\}$	$\{a0, b1, d0, d1, y2\}$	$\{a_1, a_7, a_{11}, a_{12}, a_{35}\}$	$\{a0, b1, d0, d1, y3\}$
9	$\{\varnothing\}$	$\{a0, b1, d0, d1, y3\}$	$\{a_1, a_7, a_{11}, a_{12}, a_{43}\}$	$\{a0, b1, d0, d1, y4\}$
10	$\{\varnothing\}$	$\{a0, b1, d0, d1, y4\}$	$\{a_1, a_7, a_{11}, a_{12}, a_{51}\}$	$\{a0, b1, d0, d1, Z\}$
11	$\{\varnothing\}$	$\{a0, b1, d0, d1, Z\}$	$\{a_{53}, a_{57}, a_{58}, a_{62}\}$	$\{a1, b0, b1, y0\}$
12	$\{\varnothing\}$	$\{a1, b0, b1, y0\}$	$\{a_2, a_6, a_7, a_{17}\}$	$\{a1, b0, b1, d0, y1\}$
13	$\{\varnothing\}$	$\{a1, b0, b1, d0, y1\}$	$\{a_2, a_6, a_7, a_{11}, a_{22}\}$	$\{a1, b0, b1, d0, c2, y2\}$
14	$\{\varnothing\}$	$\{a1, b0, b1, d0, c2, y2\}$	$\{a_2, a_6, a_7, a_{11}, a_{34}\}$	$\{a1, b0, b1, d0, d2, y3\}$
15	$\{\varnothing\}$	$\{a1, b0, b1, d0, d2, y3\}$	$\{a_2, a_6, a_7, a_{11}, a_{13}, a_{43}\}$	$\{a1, b0, b1, d0, d2, y4\}$
16	$\{\varnothing\}$	$\{a1, b0, b1, d0, d2, y4\}$	$\{a_2, a_6, a_7, a_{11}, a_{13}, a_{51}\}$	$\{a1, b0, b1, d0, d2, Z\}$
17	$\{\varnothing\}$	$\{a1, b0, b1, d0, d2, Z\}$	$\{a_{52}, a_{53}, a_{57}, a_{59}, a_{62}\}$	$\{a0, a1, b0, b2, y0\}$
18	$\{\varnothing\}$	$\{a0, a1, b0, b2, y0\}$	$\{a_1, a_2, a_6, a_8, a_{18}\}$	$\{a0, a1, b0, b2, c1, y1\}$
19	$\{\varnothing\}$	$\{a0, a1, b0, b2, c1, y1\}$	$\{a_1, a_2, a_6, a_8, a_{23}\}$	$\{a0, a1, b0, b2, c2, y2\}$
20	$\{\varnothing\}$	$\{a0, a1, b0, b2, c2, y2\}$	$\{a_1, a_2, a_6, a_8, a_{32}\}$	$\{a0, a1, b0, b2, c3, y3\}$
21	$\{\varnothing\}$	$\{a0, a1, b0, b2, c3, y3\}$	$\{a_1, a_2, a_6, a_8, a_{42}\}$	$\{a0, a1, b0, b2, d3, y4\}$
22	$\{\varnothing\}$	$\{a0, a1, b0, b2, d3, y4\}$	$\{a_1, a_2, a_6, a_8, a_{14}, a_{51}\}$	$\{a0, a1, b0, b2, d3, Z\}$
23	$\{\varnothing\}$	$\{a0, a1, b0, b2, d3, Z\}$	$\{a_{52}, a_{54}, a_{60}, a_{62}\}$	$\{a0, a2, b3, y0\}$
24	$\{\varnothing\}$	$\{a0, a2, b3, y0\}$	$\{a_1, a_3, a_9, a_{16}\}$	$\{a0, a2, b3, d0, y1\}$
25	$\{\varnothing\}$	$\{a0, a2, b3, d0, y1\}$	$\{a_1, a_3, a_9, a_{11}, a_{27}\}$	$\{a0, a2, b3, d0, y2\}$

26	$\{\varnothing\}$	$\{a0, a2, b3, d0, y2\}$	$\{a_1, a_3, a_9, a_{11}, a_{28}\}$	$\{a0, a2, b3, d0, d2, y3\}$
27	$\{\varnothing\}$	$\{a0, a2, b3, d0, d2, y3\}$	$\{a_1, a_3, a_9, a_{11}, a_{13}, a_{37}\}$	$\{a0, a2, b3, d0, d2, d3, y4\}$
28	$\{\varnothing\}$	$\{a0, a2, b3, d0, d2, d3, y4\}$	$\{a_1, a_3, a_9, a_{11}, a_{13}, a_{14}, a_{51}\}$	$\{a0, a2, b3, d0, d2, d3, Z\}$
29	$\{\varnothing\}$	$\{a0, a2, b3, d0, d2, d3, Z\}$	$\{a_{55}, a_{57}, a_{59}, a_{60}, a_{62}\}$	$\{a3, b0, b2, b3, y0\}$
30	$\{\varnothing\}$	$\{a3, b0, b2, b3, y0\}$	$\{a_4, a_6, a_8, a_9, a_{17}\}$	$\{a3, b0, b2, b3, d0, y1\}$
31	$\{\varnothing\}$	$\{a3, b0, b2, b3, d0, y1\}$	$\{a_4, a_6, a_8, a_9, a_{11}, a_{27}\}$	$\{a3, b0, b2, b3, d0, y2\}$

Bibliography

[1] Azimi, S., Iancu, B., and Petre, I. Reaction system models for the heat shock response. *Fundamenta Informaticae 131* (2014), 299–312.

[2] Brayton, R., Hachtel, G., McMullen, C., and Sangiovanni-Vincentelli, A. The espresso-ii minimization loop and algorithms. 54–138.

[3] Brijder, R., Ehrenfeucht, A., Main, M., and Rozenberg, G. A tour of reaction systems. *Int. J. Found. Comput. Sci. 22* (2011), 1499–1517.

[4] Corolli, L., Maj, C., Marini, F., Besozzi, D., and Mauri, G. An excursion in reaction systems: from computer science to biology. *Theoretical Computer Science 454* (2012), 95–108.

[5] Ehrenfeucht, A., Kleijn, J., Koutny, M., and Rozenberg, G. Minimal reaction systems. *Transactions on Computational Systems Biology XIV. Lecture Notes in Computer Science 7625* (2012).

[6] Ehrenfeucht, A., Main, M., and Rozenberg, G. Combinatorics of life and death for reaction systems. *Int. J. Found. Comput. Sci. 21* (2010), 345–356.

[7] Ehrenfeucht, A., Main, M., and Rozenberg, G. Functions defined by reaction systems. *Int. J. Found. Comput. Sci. 22* (2011), 167–178.

[8] Ehrenfeucht, A., and Rozenberg, G. Reaction systems. *Fundamenta Informaticae 75*, 1-4 (2007), 263–280.

[9] Formenti, E., Manzoni, L., and Porreca, A. On the complexity of occurrence and convergence problems in reaction systems. *Natural Computing 14* (2014), 185–191.

[10] Genova, D., Hoogeboom, H., and Jonoska, N. A graph isomorphism condition and equivalence of reaction systems. *Theoretical Computer Science 701* (2017).

[11] Genova, D., Hoogeboom, H., and Prodanoff, Z. Extracting reaction systems from function behavior. *Journal of Membrane Computing 2* (2020), 1–13.

[12] Ivanov, S. brsim: Basic reaction system simulator. https://github.com/scolobb/brsim.

[13] Riznyk, V., and Solomko, M. Minimization of boolean functions by combinatorial method. *Technology audit and production reserves 4* (2017), 49–64.

[14] Rudell, R. Espresso(1) manual page. http://bear.ces.cwru.edu/eecs_cad/man_octtools_espresso.html, 1993.

[15] Salomaa, A. Functions and sequences generated by reaction systems. *Theoretical Computer Science 466* (2012), 87–96.

[16] Salomaa, A. On state sequences defined by reaction systems. *Logic and Program Semantics. Lecture Notes in Computer Science 7230* (2012), 271–282.

[17] Salomaa, A. Functional constructions between reaction systems and propositional logic. *International Journal of Foundations of Computer Science 24* (2013).

[18] Salomaa, A. Minimal and almost minimal reaction systems. *Natural Computing 12*, 3 (2013), 369–376.

[19] Salomaa, A. Compositions of reaction systems. *J. Autom. Lang. Comb. 19* (2014), 279–290.

[20] Salomaa, A. Minimal reaction systems defining subset functions. *Computing with New Resources. Lecture Notes in Computer Science.* (2014), 436–446.

[21] Salomaa, A. Two-step simulations of reaction systems by minimal ones. *Acta Cybernetica 22* (2015), 247–257.