The Number of Languages with Maximum State Complexity

Bjørn Kjos-Hanssen(✉) and Lei Liu

University of Hawaii at Manoa, Honolulu, USA
bjoernkh@hawaii.edu

Abstract. Champarnaud and Pin (1989) found that the minimal deterministic automaton of a language $L \subset \Sigma^n$, where $\Sigma = \{0, 1\}$, has at most

$$\sum_{i=0}^{n} \min(2^i, 2^{2^{n-i}} - 1)$$

states, and for each n there exists L attaining this bound. Câmpeanu and Ho (2004) have shown more generally that the tight upper bound for Σ of cardinality k and for complete automata is

$$\frac{k^r - 1}{k - 1} + \sum_{j=0}^{n-r}(2^{k^j} - 1) + 1$$

where $r = \min\{m : k^m \geq 2^{k^{n-m}} - 1\}$. (In these results, requiring totality of the transition function adds 1 to the state count.) Câmpeanu and Ho's result can be viewed as concerning functions $f : [k]^n \to [2]$ where $[k] = \{0, \ldots, k-1\}$ is a set of cardinality k. We generalize their result to arbitrary function $f : [k]^n \to [c]$ where c is a positive integer.

Let O_i be the number of functions from $[b^i]$ to $[c^{b^{n-i}}]$ that are onto $[c^{b^{n-i}} - 1]$. Câmpeanu and Ho stated that it is very difficult to determine the number of maximum-complexity languages. Here we show that it is equal to O_i, for the least i such that $O_i > 0$.

For monotone languages a tightness result seems harder to obtain. However, we show that the following upper bound is attained for all $n \leq 10$.

$$\sum_{i=0}^{n} \min(2^i, M(n-i) - 1),$$

where $M(k)$ is the kth Dedekind number.

1 Introduction

The function $+$ on $\mathbb{Z}/5\mathbb{Z}$ may seem rather complicated as functions on that set go. On the other hand, $f(x, y, z) = x + y + z \bmod 5$ is less so, in that we can decompose it as $(x + y) + z$, so that after seeing x and y, we need not remember the pair (x, y) but only their sum. Out of the 5^{5^3} ternary functions

T. V. Gopal and J. Watada (Eds.): TAMC 2019, LNCS 11436, pp. 394–409, 2019.
https://doi.org/10.1007/978-3-030-14812-6_24

on a 5-element set, at most $5^{2\cdot 5^2}$ can be decomposed as $(x *_1 y) *_2 z$ for some binary functions $*_1$, $*_2$. In Sect. 2 we make precise a sense in which such are not the most complicated ternary functions. We do this by extending a result of Câmpeanu and Ho [3] to functions taking values in a set of size larger than two.

Rising to an implicit challenge posed by Câmpeanu and Ho, we give a formula for the number of maximally complex languages in Sect. 2.2.

A motivation from finance will be felt in Sects. 3 and 4. The complexity of financial securities came into focus with the 2008 financial crisis. While Arora et al. [1] obtained NP-hardness results for the pricing of a security, here we look at the automatic complexity associated with executing a given trading strategy. The possibility of exercising early leads to a less complex option in our sense, as is easy to see. Thus we shall restrict attention to options which are European insofar as they can only be exercised at the final time n.

2 Complexity of Languages and Operations

Definition 2.1. A deterministic finite automaton (DFA) [9] M is a 5-tuple, $(Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states,
- $q_0 \in Q$ is the start state,
- $F \subseteq Q$ is the set of accept states,
- Σ is a finite set of input symbols and
- $\delta : Q \times \Sigma \longrightarrow Q$ is the transition function.

If δ is not required to be total then we speak of a *partial deterministic finite automaton (PDFA).*

Definition 2.2. Let $\Sigma = \{0, 1\}$, let $n \in \mathbb{Z}^+$ and $X \subseteq \Sigma^{\leq n}$. Define $A_-(X)$ to be the minimum $|Q|$ over all PDFAs $M = (Q, \Sigma, \delta, q_0, F)$ for which $L(M)$, the language recognized by M, equals X. We call a PDFA $M = (Q, \Sigma, \delta, q_0, F)$ *minimal for* X if

$$|Q| = A_-(X).$$

2.1 Operations

Champarnaud and Pin [4] obtained the following result.

Theorem 2.3 ([4, Theorem 4]). *The minimal PDFA of a language $L \subset \{0, 1\}^n$ has at most*

$$\sum_{i=0}^{n} \min(2^i, 2^{2^{n-i}} - 1)$$

states, and for each n there exists L attaining this bound.

Theorem 2.3 was generalized by Câmpeanu and Ho [3]:

Theorem 2.4 ([3, Corollary 10])**.** *For $k \geq 1$, let $[k] = \{0, \dots, k-1\}$. Let $l \in \mathbb{N}$ and let M be a minimal DFA for a language $L \subseteq [k]^l$. Let Q be the set of states of M. Then we have:*

(i) $\#Q \leq \frac{k^r-1}{k-1} + \sum_{j=0}^{l-r}(2^{k^j}-1)+1$, *where* $r = \min\{m \mid k^m \geq 2^{k^{l-m}}-1\}$*;*
(ii) there is an M such that the upper bound given by (i) is attained.

Both of these results involve an upper bound which can be viewed as a special case of Theorem 2.7 below.

Definition 2.5. Let b, n, and c be positive integers. We say that a PDFA M *accepts* a function $f : [b]^n \to [c]$ if there are $c-1$ many special states $q_1, \dots, q_{c-1}$ of M such that for all $\vec{x} \in [b]^n$,

- for $i > 0$, $f(\vec{x}) = i$ iff M on input $\vec{x}$ ends in state q_i; and
- $f(\vec{x}) = 0$ iff M does not end in any of the special states on input $\vec{x}$.

Definition 2.5 generalizes the case $b = 2$ studied by Champarnaud and Pin. We write A^B for the set of all functions from B to A.

Definition 2.6. Let $[c]^{[b]^n}$ be the set of n-ary functions $f : [b]^n \to [c]$. Let b and c be positive integers and let $\mathfrak{C} \subseteq [c]^{[b]^n}$. The *Champarnaud–Pin family* of $\mathfrak{C}$ is the family of sets $\{\mathfrak{C}_k\}_{0 \leq k \leq n}$, where $\mathfrak{C}_k \subseteq [c]^{[b]^{n-k}}$, $0 \leq k \leq n$, given by

$$\mathfrak{C}_k = \{g \in [c]^{[b]^{n-k}} : \exists f \in \mathfrak{C},\, \vec{d} \in [b]^k \quad \forall \vec{x} \quad g(\vec{x}) = f(\vec{d}, \vec{x})\}.$$

So $\mathfrak{C}_0 = \mathfrak{C}$, $\mathfrak{C}_1$ is obtained from $\mathfrak{C}_0$ by plugging in constants for the first input, and so forth. We write $\mathfrak{C}_n^- = \{f \in \mathfrak{C}_n : f \neq 0\}$ in order to throw out the constant zero function. Note that $|\mathfrak{C}_n^-| \geq |\mathfrak{C}_n| - 1$.

Theorem 2.7. *Let b and c be positive integers. Let $\mathfrak{C} \subseteq [c]^{[b]^n}$. An upper bound on the minimal number of states of PDFAs accepting members of $\mathfrak{C}$ is given by*

$$\sum_{i=0}^{n} \min(b^i, |\mathfrak{C}_i^-|).$$

The proof will be apparent from the proof of the next result, which is a generalization of Câmpeanu and Ho's theorem.

Theorem 2.8. *Let b and c be positive integers. For the minimal number of states of PDFAs M accepting functions $f : [b]^n \to [c]$, the upper bound*

$$\sum_{i=0}^{n} \min(b^i, c^{b^{n-i}} - 1)$$

is attained.

Proof. Let $\log = \log_b$. The critical point for this result is the pair of values (i, k) with $i + k = n$ such that $b^i \le c^{b^k} - 1$ (i.e., $i < b^k \log c$) and $b^{i+1} > c^{b^{k-1}} - 1$ (i.e., $b^{i+1} \ge c^{b^{k-1}}$, i.e., $(i+1) \ge b^{k-1} \log c$), which can be summarized as

$$b^{k-1} \log c - 1 \le i < b^k \log c. \tag{1}$$

$$b^{k-1} \log c \le i + 1 \le b^k \log c.$$

We shall define a set A of k-ary functions of size $(c^{b^{k-1}})/b$ which when using the b many transitions (substitutions for say p_1) maps onto each of the $c^{b^{k-1}}$ many $k-1$-ary functions α. This will suffice if

$$c^{b^{k-1}}/b \le b^i$$

which does hold for all b by (1). The construction is similar to that of [3, Figure 1 and Theorem 8]; we shall be slightly more explicit than they were. Let $s = c^{b^{k-1}} - 1$. Let $f_0, \dots, f_{s-1}$ the set of all nonzero $k-1$-ary functions. As s may not be divisible by b, let us write $s = qb + r$ with quotient $q \ge 0$ and remainder $0 \le r < b$. For j with $0 \le j \le q - 1$, let g_j be given by

$$g_j(i, \vec{x}) = f_{jb+i}(\vec{x})$$

for each $i \in [b]$ and $\vec{x} \in [b]^{k-1}$. Let g_q be given by $g_q(i, \vec{x}) = f_{qb+i}(\vec{x})$ for each $0 \le i \le r - 1$, and let $g_q(i, \vec{x})$ be arbitrary for $r \le i < b$. Finally, extend the set of functions $g_0, \dots, g_q$ to b^i many k-ary functions in an arbitrary way, obtaining functions h_σ for $\sigma \in [b]^i$. Then our function attaining the bound is given by

$$H(\sigma, \tau) = h_\sigma(\tau).$$

$\square$

When $b = 2$ and c is larger, Theorem 2.8 corresponds to automatic complexity of equivalence relations on binary strings as studied in [6]. When $b = c$, we have the case of n-ary operations on a given finite set, which is of great interest in universal algebra.

2.2 The Number of Maximally Complex Languages

Definition 2.9. Let b and c be positive integers and let $0 \le i \le n$. Let $O_i = O_i^{(b,c,n)}$ be the number of functions from $[b^i]$ to $[c^{b^{n-i}}]$ that are onto $[c^{b^{n-i}} - 1]$. That is, functions $f : [b^i] \to [c^{b^{n-i}}]$ such that for each $y \in [c^{b^{n-i}} - 1]$ there is an $x \in [b^i]$ with $f(x) = y$.

Câmpeanu and Ho lamented that it seemed very difficult to count the number of maximum-complexity languages. Here we show

Theorem 2.10. *Let b and c be positive integers and let $n \geq 0$. The number of maximum complexity functions $f : [b]^n \to [c]$ is O_i, where $0 \leq i \leq n$ is minimal such that $O_i > 0$.*

Proof. Champarnaud and Pin, and Câmpeanu and Ho, and the present authors in Theorem 2.8, all found a maximal complexity by explicitly exhibiting the general automaton structure of a maximal-complexity language: we start with states corresponding to binary strings and end with strings corresponding to Boolean functions, and there is a crossover point in the middle where, in order that all states be used, we need an onto function exactly as specified in the definition of O_i. The crossover point occurs for the least i such that $O_i > 0$, which is when the value of the minimum of $(b^i, c^{b^{n-i}} - 1)$ switches from the first to the second coordinate. The number of such functions is then the number of such onto functions. Since we do not require totality and do not use a state for output 0 ("reject") we omit the constant 0 Boolean function in the range of our onto maps. □

Note that the number of onto functions is well known in terms of Stirling numbers of the second kind. Let $O_{m,n}$ be the number of onto functions from $[m]$ to $[n]$. Then

$$O_{m,n} = n! \left\{ {m \atop n} \right\},$$

where $\left\{ {m \atop n} \right\}$, the number of equivalence relations on $[m]$ with n equivalence classes, is a Stirling number of the second kind.

Note also that the number of functions from $[a]$ to $[b]$ that are onto the first $b - 1$ elements of $[b]$ is, in terms of the number m of elements going to the not-required element,

$$\sum_{m=0}^{a-(b-1)} \binom{a}{m} O_{a-m,b-1}.$$

Example 2.11. When $n = 3$ and $b = c = 2$, we have that O_i is the number of functions from 2^i to $2^{2^{3-i}}$ that are onto $2^{2^{3-i}} - 1$. In this case, $O_1 = 0$. However, O_2 is the number of functions from 4 to 4 that are onto 3. This is

$$\sum_{m=0}^{4-(4-1)} \binom{4}{m} O_{4-m,4-1} = O_{4,3} + 4\,O_{3,3} = 36 + 24 = 60.$$

These 60 languages are shown in Table 1.

Table 1. All possible sets Z with $A_{-}(Z) = 7$.

Size	Z
$\|Z\| = 4$	$\{000, 001, 010, 101\}$, $\{000, 001, 010, 111\}$,
	$\{000, 001, 011, 100\}$, $\{000, 001, 100, 111\}$,
	$\{000, 001, 011, 110\}$, $\{000, 001, 101, 110\}$,
	$\{000, 010, 011, 101\}$, $\{000, 010, 011, 111\}$,
	$\{001, 010, 011, 100\}$, $\{010, 011, 100, 111\}$,
	$\{001, 010, 011, 110\}$, $\{010, 011, 101, 110\}$,
	$\{000, 011, 100, 101\}$, $\{000, 100, 101, 111\}$,
	$\{001, 010, 100, 101\}$, $\{010, 100, 101, 111\}$,
	$\{001, 100, 101, 110\}$, $\{011, 100, 101, 110\}$;
	$\{000, 011, 110, 111\}$, $\{000, 101, 110, 111\}$,
	$\{001, 010, 110, 111\}$, $\{010, 101, 110, 111\}$,
	$\{001, 100, 110, 111\}$, $\{011, 100, 110, 111\}$
$\|Z\| = 5$	$\{000, 001, 010, 100, 111\}$, $\{000, 001, 010, 101, 110\}$,
	$\{000, 001, 011, 100, 110\}$, $\{000, 001, 011, 101, 110\}$,
	$\{000, 001, 011, 100, 111\}$, $\{000, 001, 010, 101, 111\}$,
	$\{000, 010, 011, 100, 111\}$, $\{000, 010, 011, 101, 110\}$,
	$\{001, 010, 011, 100, 110\}$, $\{001, 010, 011, 101, 110\}$,
	$\{001, 010, 011, 100, 111\}$, $\{000, 010, 011, 101, 111\}$,
	$\{000, 010, 100, 101, 111\}$, $\{000, 011, 100, 101, 110\}$,
	$\{001, 010, 100, 101, 110\}$, $\{001, 011, 100, 101, 110\}$,
	$\{001, 010, 100, 101, 111\}$, $\{000, 011, 100, 101, 111\}$,
	$\{000, 010, 101, 110, 111\}$, $\{000, 011, 100, 110, 111\}$,
	$\{001, 010, 100, 110, 111\}$, $\{001, 011, 100, 110, 111\}$,
	$\{001, 010, 101, 110, 111\}$, $\{000, 011, 101, 110, 111\}$
$\|Z\| = 6$	$\{000, 001, 010, 011, 100, 111\}$,
	$\{000, 001, 010, 011, 101, 110\}$,
	$\{000, 001, 010, 100, 101, 111\}$,
	$\{000, 001, 011, 100, 101, 110\}$,
	$\{000, 001, 010, 101, 110, 111\}$,
	$\{000, 001, 011, 100, 110, 111\}$,
	$\{000, 010, 011, 100, 101, 111\}$,
	$\{001, 010, 011, 100, 101, 110\}$,
	$\{000, 010, 011, 101, 110, 111\}$,
	$\{001, 010, 011, 100, 110, 111\}$,
	$\{000, 011, 100, 101, 110, 111\}$,
	$\{001, 010, 100, 101, 110, 111\}$

Listing 1. Pseudocode for our variant of the Myhill–Nerode algorithm.

```
Input: Strings s and t, a set of strings L, and a max length n.
Output: The boolean of whether s and t are equivalent for L.
For u a binary string of length between 0 and n−1,
  if len(s+u), len(t+u) both at most n
  and exactly one of s+u, t+u is in the up−closure of L,
    return False
Return True.
```

2.3 Polynomial-Time Algorithm

It is perhaps worth pointing out that there is a polynomial-time algorithm for finding the minimal automaton of Boolean functions, based on essentially the Myhill–Nerode theorem [5,10]. In this subsection we detail that somewhat.

Definition 2.12. Given a language L, and a pair of strings x and y, define a *distinguishing extension* to be a string z such that exactly one of the two strings xz and yz belongs to L. Define a relation R_L on strings by the rule that xR_Ly if there is no distinguishing extension for x and y.

As is well known, R_L is an equivalence relation on strings, and thus it divides the set of all strings into equivalence classes.

Theorem 2.13 (Myhill–Nerode). *A language L is regular if and only if R_L has a finite number of equivalence classes. Moreover, the number of states in the smallest deterministic finite automaton (DFA) recognizing L is equal to the number of equivalence classes in R_L. In particular, there is a unique DFA with minimum number of states.*

The difference is that for us we require $|xz| \leq n$ and $|yz| \leq n$, see Listing 1.

3 Monotone Boolean Functions

The main theoretical results of the paper are in Sect. 2. The present, longer section deals with a more computational and exploratory investigation: what happens if we try to prove that the natural upper bound on complexity is attained in restricted settings such as monotone functions?

Definition 3.1. An isotone map is a function φ with $a \leq b \implies \varphi(a) \leq \varphi(b)$.

The Online Encyclopedia of Integer Sequences (OEIS) has a tabulation of Dedekind numbers, i.e., the number $M(n)$ of monotone functions [12], which is also the number of elements of the free distributive lattice on n generators and the number of antichains of subsets of $[n]$.

Definition 3.2. For an integer $n \geq 0$, F_n is the set of monotone Boolean functions of n variables (equivalently, the free distributive lattice on n generators, allowing 0 and 1 to be included), and $F_n^- = F_n \setminus \{0\}$ where 0 is the constant 0 function.

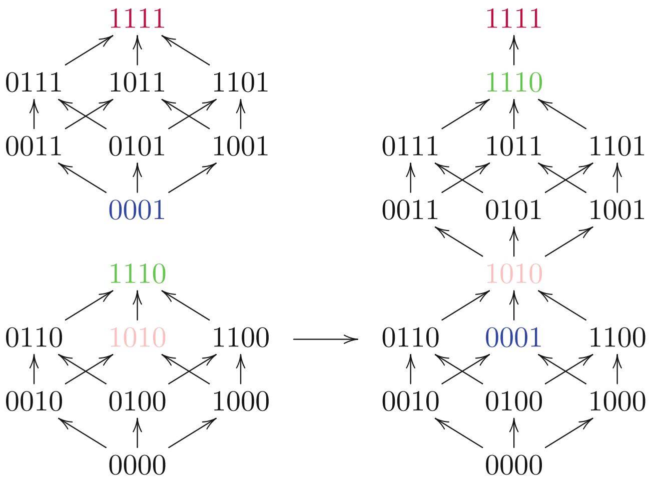

Fig. 1. Isotone 1:1 map from $\mathbf{2}^4$ to F_3^-.

The illustrative case $n = 3$ is shown in Fig. 4.

Theorem 3.3. *The minimal automaton of a monotone language $L \subset \{0,1\}^n$ has at most*

$$\sum_{i=0}^{n} \min(2^i, |F_{n-i}| - 1)$$

states. This bound is attained for $n \leq 10$.

Proof. The upper bound follows from Theorem 2.7. The sharpness results are obtained in a series of theorems tabulated in Table 2. □

Thinking financially, an option is monotone if whenever s is pointwise dominated by t and $s \in L$ then $t \in L$, where L is the set of exercise situations for the option. This is the case for common options like call options or Asian average-based options and makes financial sense if a rise in the underlying is always desirable and always leads to a higher option value.

Example 3.4 (Asian option; Shreve [11, Exercise 1.8])**.** This is the example that in part motivates our looking at monotone options. Let $n = 3$ and consider a starting capital $S_0 = 4$, up-factor $u = 2$, down-factor $d = \frac{1}{2}$. Let $Y_i = \sum_{k=0}^{i} S_k$. The payoff at time $n = 3$ is $(\frac{1}{4}Y_3 - 4)^+$. To fit this example into our framework in the present paper, let us look at which possibilities lead to exercising, i.e., $\frac{1}{4}Y_3 - 4 > 0$ or $Y_3 > 16$. Computation shows that the set of exercise outcomes is $\{011, 100, 101, 110, 111\}$. The complexity is 6 (Fig. 3), so it is maximally complex for a monotone option.

For $n = 3$ we are looking at isotone functions from $\{0,1\}$ to the family of monotone functions on two variables p and q. For the Asian option in Example 3.4 $\{0,1\}$ are mapped to $\{p \wedge q, 1\}$. For the majority function, $\{0,1\}$ are mapped to $\{p \wedge q, p \vee q\}$ (Fig. 3).

Table 2. Maximum complexity of monotone securities.

n	Adequacy diagram	#States	Proof/witness
0	$\begin{tikzcd} 1 \arrow[d] \\ (1) \end{tikzcd}$	1	
1	$\begin{tikzcd} 1 \arrow[d] & & (2) \\ (2) & \rightrightarrows & 1 \end{tikzcd}$	2	Theorem 3.9
2	$\begin{tikzcd} 1 & 2 \arrow[d] & & (4) \\ & (2) & \rightrightarrows & 1 \end{tikzcd}$	4	Theorem 3.9
3	$\begin{tikzcd} 1 & 2 \arrow[d] & & (4) & \\ & (5) & \rightrightarrows & 2 & 1 \end{tikzcd}$	6	Theorem 3.9; Example 3.4
4	$\begin{tikzcd} 1 & 2 & 4 \arrow[d] & & (8) & \\ & & (5) & \rightrightarrows & 2 & 1 \end{tikzcd}$	10	Theorem 3.9
5	$\begin{tikzcd} 1 & 2 & 4 \arrow[d] & & (8) & & \\ & & (19) & \rightrightarrows & 5 & 2 & 1 \end{tikzcd}$	15	Theorem 3.9
6	$\begin{tikzcd} 1 & 2 & 4 & 8 \arrow[d] & & (16) & & \\ & & & (19) & \rightrightarrows & 5 & 2 & 1 \end{tikzcd}$	23	Theorem 3.10
7	$\begin{tikzcd} 1 & 2 & 4 & 8 & 16 \arrow[d] & & (32) & & \\ & & & & (19) & \rightrightarrows & 5 & 2 & 1 \end{tikzcd}$	39	Figure 1; Theorem 3.10
8	$\begin{tikzcd} 1 & 2 & 4 & 8 & 16 \arrow[d] & & (32) & & & \\ & & & & (167) & \rightrightarrows & 19 & 5 & 2 & 1 \end{tikzcd}$	58	Theorem 3.8
9	$\begin{tikzcd} 1 & 2 & 4 & 8 & 16 & (32) \arrow[d] & & (64) & & & \\ & & & & & (167) & \rightrightarrows & 19 & 5 & 2 & 1 \end{tikzcd}$	90	Theorem 3.13
10	$\begin{tikzcd} 1 & 2 & 4 & 8 & 16 & 32 & 64 \arrow[d] & & (128) & & & \\ & & & & & & (167) & \rightrightarrows & 19 & 5 & 2 & 1 \end{tikzcd}$	154	Theorem 3.12

The sets $\{p \wedge q, p \vee q\}$ and $\{p \wedge q, 1\}$ both have the desirable property (from the point of view of increasing the complexity) that by substitution we obtain a full set of nonzero monotone functions in one fewer variables, in this case $\{p, 1\}$.

Definition 3.5. Let us say that a set of monotone functions on variables $p_1, \ldots, p_n$ is *adequate* if by substitutions of values for $p_1 \in \{0, 1\}$ they contain all monotone nonzero functions on $p_2, \ldots, p_n$. If one value for p_1 suffices then we say *strongly adequate*.

Let us write $\mathbf{2}^i$ for the set $\{0, 1\}^i$ with the product ordering.

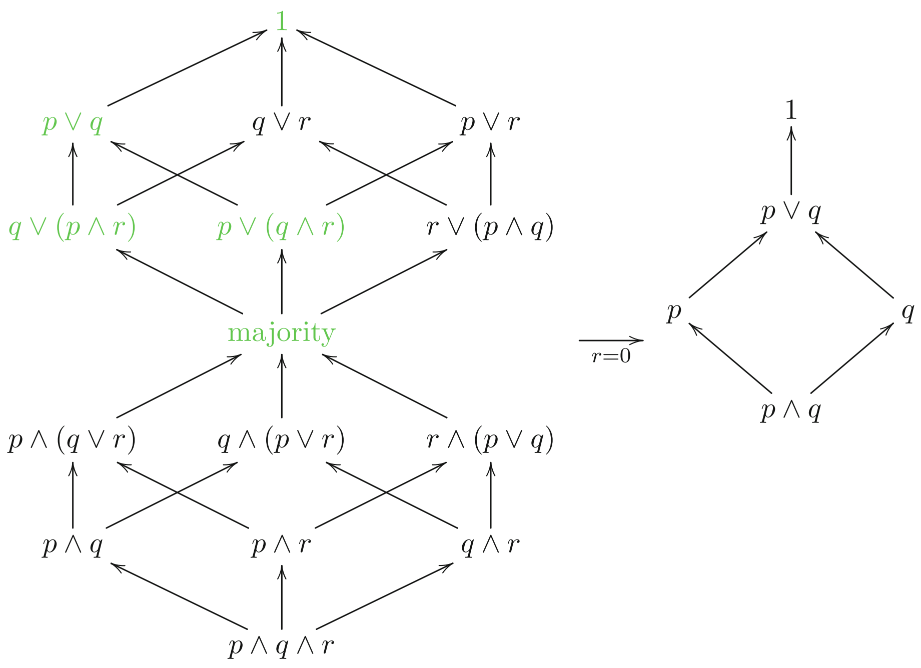

Fig. 2. Adequacy in the proof that $2^4 \to 19 \rightrightarrows 5$.

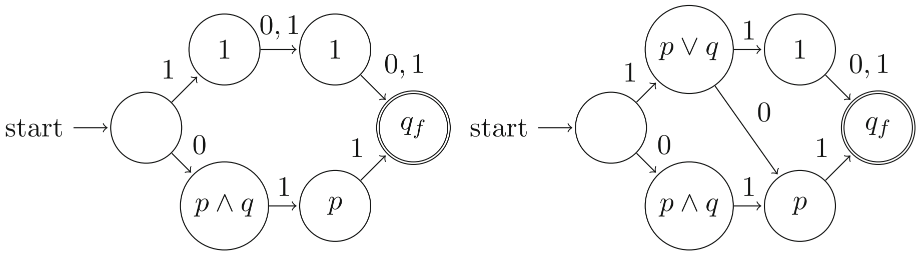

Fig. 3. Asian option, and European call option (corresponding to the majority function).

Definition 3.6. If there is an embedding of $\mathbf{2}^i$ into F_j^- ensuring adequacy onto F_{j-1}^-, in the sense that we map *into* F_{j-1} (so self-loops may be used in the automaton), and we map *onto* F_{j-1}^-, then we write

$$2^i \to |F_j^-| \rightrightarrows |F_{j-1}^-|.$$

It is crucial to note that in Sect. 2, adequacy was automatic: the concept of function is much more robust than that of a monotone function, meaning that functions can be combined in all sorts of ways and remain functions. As an example of the unusual but convenient notation of Definition 3.6, we have:

Theorem 3.7. *There is an embedding of* $\mathbf{2}^2$ *into* F_3^- *ensuring adequacy onto* F_2^-*. In symbols,*

$$4 \to 19 \rightrightarrows 5.$$

Proof. We use formulas of the form $(r \wedge b) \vee a$ with $a \leq b$, as follows:

$\mathbf{2}^2$	F_3^-				F_2^-
$(0,1)$	$(r \wedge p)$	$\vee$	p	$\equiv$	p
$(1,0)$	$(r \wedge q)$	$\vee$	q	$\equiv$	q
$(0,0)$	$(r \wedge (p \wedge q))$	$\vee$	$(p \wedge q)$	$\equiv$	$p \wedge q$
$(1,1)$	$(r \wedge 1)$	$\vee$	$(p \vee q)$	$\mapsto_{r=1}$	1
				$\mapsto_{r=0}$	$p \vee q$

□

Theorem 3.8. *There is an embedding of* $\mathbf{2}^4$ *into* F_4^- *ensuring adequacy onto* F_3^-*:*

$$16 \to 167 \rightrightarrows 19$$

Proof. We make sure to hit p, q, r as follows: $(r \wedge b) \vee a_i$, $1 \leq i \leq 2$, where $a_1 < a_2 \leq b$, and $a_i, b \in F_3^-$, with $b \in T$. Here T is the top cube in F_3^-,

$$T = \{b \in F_3^- : \text{maj} \leq b \leq p \vee q \vee r\}$$

$$= \left\{ \begin{array}{l} (p \wedge q) \vee (p \wedge r) \vee (q \wedge r), \\ p \vee (q \wedge r), \quad q \vee (p \wedge r), \quad r \vee (p \wedge q), \\ p \vee q, \quad p \vee r, \quad q \vee r, \quad p \vee q \vee r \end{array} \right\}.$$

Let $\psi : \{0,1\}^3 \to T$ be an isomorphism. Not that $b \notin \{\hat{0}, p, q, r\}$. And $\{a_1, a_2\} \subset \{b, p, q, r, \hat{0}\}$ where $\hat{0}$ is $p \wedge q \wedge r$, the least element of F_3^-. Let

$$(a_1, a_2) = \begin{cases} (\hat{0}, b) & \text{if } b \text{ bounds none of } p, q, r; \\ (p, b) \text{ or } (\hat{0}, p) & \text{if say } b > p; \end{cases}$$

By Lemma above, $(r \wedge \psi(x)) \vee a_i \leq (r \wedge \psi(y)) \vee c_i$ iff $x \leq y$ and $a_i \leq c_i$. □

We can consider whether $u \to v \rightrightarrows w$ whenever the numbers are of the form 2^m, $|F_n^-| \in \{1, 2, 5, 19, 167, \dots\}$, $|F_{n-1}^-|$, and $u \leq v$ and $w \leq 2u$ (as u increases, being 1:1 becomes harder but being adequate becomes easier). In the case of strong adequacy witnessed by $p = p_0$ we write simply $u \to v \to_{p_0} w$; this can only happen when $w \leq u$.

Theorem 3.9. *We have the following adequacy calculations:*

1. $2^0 \to 2 \to 1$
2. $2^1 \to 2 \to 1$
3. $2^0 \to 5 \rightrightarrows 2$
4. $2^1 \to 5 \to 2$
5. $2^2 \to 5 \to 2$

We omit the trivial proof of Theorem 3.9.

Theorem 3.10. $2^3 \to 19 \rightrightarrows 5$ *and* $2^4 \to 19 \rightrightarrows 5$.

Proof. The map in Fig. 1 is onto $F_3^- \setminus \{p,q,r\}$ so it works. As shown in Fig. 2, if we restrict that map to the top cube, mapping onto $T \cup \{1\} \setminus \{p \vee q \vee r\}$, and set $r = 0$ then we map onto F_2^-. □

Lemma 3.11. *Let a_1, a_2, b_1, b_2 be Boolean functions of p, q, r and let*

$$f_{a_i b_i}(p,q,r,s) = [s \wedge b_i] \vee [\neg s \wedge a_i].$$

Then $f_{a_1 b_1} \le f_{a_2 b_2} \iff a_1 \le a_2$ and $b_1 \le b_2$.

Proof. By definition,

$$f_{a_1 b_1} \le f_{a_2 b_2} \iff [p_4 \wedge b_1] \vee [\neg p_4 \wedge a_1] \le [p_4 \wedge b_2] \vee [\neg p_4 \wedge a_2].$$

Clearly, $a_1 \le a_2$ and $b_1 \le b_2$ implies this, so we just need the converse. If $a_1 \not\le a_2$ then any assignment that makes p_4 false, a_1 true, and a_2 false will do. Similarly if $b_1 \not\le b_2$ then any assignment that makes p_4 true, b_1 true, and b_2 false will do. □

Theorem 3.12. *There is an injective isotone map from $\mathbf{2}^6$ into F_4, and in fact*

$$2^6 \to 167 \rightrightarrows 19.$$

Proof. We start with a monotone version of the simple equation $2^{2^n} = (2^{2^{n-1}})^2$. Namely, a pair of monotone functions g, h of $n-1$ variables, with $g \le h$, gives another monotone function via

$$\begin{aligned} f(p_1,\ldots,p_n) &= [p_n \wedge f(p_1,\ldots,p_{n-1},1)] \vee [\neg p_n \wedge f(p_1,\ldots,p_{n-1},0)] \\ &= [p_n \wedge h(p_1,\ldots,p_{n-1})] \vee [\neg p_n \wedge g(p_1,\ldots,p_{n-1})] \\ &= [p_n \wedge h(p_1,\ldots,p_{n-1})] \vee g(p_1,\ldots,p_{n-1}). \end{aligned}$$

Now consider elements a of the bottom hypercube in F_3 and b of the top hypercube in F_3 in Fig. 2. So we must have $a \le b$ since the bottom is below the top (and $a = b$ can happen since the two hypercubes overlap in the majority function). Let $f_{ab} = [p_4 \wedge b] \vee [\neg p_4 \wedge a]$. Since $a \le b$, f_{ab} is monotonic. By Lemma 3.11, these functions f_{ab} are ordered as $\mathbf{2}^6 = \mathbf{2}^3 \times \mathbf{2}^3$.

Finally, in order to ensure adequacy we modify this construction to reach higher in F_4^-, replacing the top cube in the lower half by a cube formed from the upper half. In more detail, consider $(r \wedge b) \vee a$ with $a \le b$ from F_3^-, where the a's are chosen from the bottom cube of F_3, and the b's from the top cube, except that when a is the top of the bottom cube we let b be the top cube with the top replaced by 1, and when a is the bottom of the bottom cube we let b be the cube

$$\{p, q, r, p \wedge q, p \wedge r, q \wedge r, p \vee q, p \vee r, q \vee r, p \wedge q \wedge r, p \vee q \vee r\}.$$

□

Theorem 3.13. $2^5 \to 167 \rightrightarrows 19$.

Proof. A small modification of Theorem 3.12; only use bottom, top and two intermediate "cubes" within the cube. $\square$

Open Problem. For $n = 11$ we need to determine whether the following holds, which has so far proved too computationally expensive:

$$2^7 \to 167 \rightrightarrows 19?$$

That is, is there an isotone map from the 128-element lattice $\mathbf{2}^7$ into F_4^-, the set of nonzero monotone functions in variables p, q, r, s, such that upon plugging in constants for p, we cover all of F_3^-, the set of nonzero monotone functions in q, r, s?

$$\begin{tikzcd}
\text{Probability} & & & & \\
1 & & 1 & & \\
7/8 & & p \vee q \vee r \arrow[u] & & \\
6/8 & q \vee r \arrow[ur] & p \vee r \arrow[u] & & p \vee q \arrow[ull] \\
5/8 & r \vee (p \wedge q) \arrow[u] \arrow[ur] & & q \vee (p \wedge r) \arrow[ull] \arrow[ur] & p \vee (q \wedge r) \arrow[ull] \arrow[u] \\
4/8 & r \arrow[u] & \text{maj} \arrow[ul] \arrow[ur] \arrow[urr] & q \arrow[u] & p \arrow[u] \\
3/8 & r \wedge (p \vee q) \arrow[u] \arrow[ur] & & q \wedge (p \vee r) \arrow[ul] \arrow[u] & p \wedge (q \vee r) \arrow[ull] \arrow[u] \\
2/8 & q \wedge r \arrow[u] \arrow[urr] & p \wedge r \arrow[ul] \arrow[urr] & & p \wedge q \arrow[ul] \arrow[u] \\
1/8 & & p \wedge q \wedge r \arrow[ul] \arrow[u] \arrow[urr] & & \\
0 & & 0 \arrow[u] & &
\end{tikzcd}$$

Fig. 4. The lattice F_3 of all monotone Boolean functions in three variables p, q, r.

4 Early-Monotone Functions and Complete Simple Games

In this section we take the financial ideas from Sect. 3 one step further, by noting that the Asian option (Example 3.4) has the added property that earlier bits matter more. In economics terms, we have what is called a *complete simple game*: there is a set of goods linearly ordered by intrinsic value. You get some of the goods and there are thresholds for how much value you need to win.

Definition 4.1. Let $e_i \in \{0,1\}^n$ be defined by $e_i(j) = 1$ if and only if $j = i$. An n-ary Boolean function f is *early* if for all $0 \le i < j < n$ and all $y \in \{0,1\}^n$ with $y(i) = y(j) = 0$, if $f(y + e_j) = 1$ then $f(y + e_i) = 1$.

The number of early (not necessarily monotone) functions starts

$$2, 4, 12, 64, 700, 36864, \ldots$$

If a function is early and monotone we shall call it early-monotone. Early-monotonicity encapsulates an idea of time-value-of-money; getting paid now is better than next week, getting promoted now is better than next decade, etc.

In the early context one needs the map from 2^m into the early functions to be "early", i.e., the function mapped to by 100 should dominate the one mapped to by 010 etc. That is, the map must be order-preserving from 2^m with the majorization lattice order into the complete simple games.

The number of early-monotone functions on n variables, including zero, is

$$2, 3, 5, 10, 27, 119, 1173, \ldots$$

which appears in OEIS A132183 as the number of "regular" Boolean functions in the terminology of Donald Knuth. He describes them also as the number of order ideals (or antichains) of the binary majorization lattice with 2^n points.

Definition 4.2. The binary majorization lattice E_n is the set $\{0,1\}^n$ ordered by $(a_1, \ldots, a_n) \le (b_1, \ldots, b_n)$ iff $a_1 + \cdots + a_k \le b_1 + \cdots + b_k$ for each k.

The lattice E_5 for $n = 5$ is illustrated in [7, Fig. 8, Volume 4A, Part 1]. The basic properties of this lattice are discussed in [7, Exercise 109 of Section 7.1.1]. The majorization order is obtained by representing e.g. 1101 as $(1, 2, 4, \infty)$, showing where the kth 1 appears (the ∞ signifying that there is no fourth 1 in 1101), and ordering these tuples by majorization. OEIS cites work of Stefan Bolus [2] who calls the "regular" functions *complete simple games* [8], a term from the economics and game theory literature. There, arbitrary monotone functions are called *simple games*, and "complete" refers to the fact that the positions have a complete linear ordering (in the finance application, earlier positions are most valuable). Figure 5 shows that in the complete-simple-games setting we have

$$\begin{tikzcd}
1 \arrow[r] & 2 \arrow[r] & 4 \arrow[r] & 8 \arrow[r] & 16 \arrow[d] & & & & \\
 & & & & 26 \arrow[r, shift left] \arrow[r, shift right] & 9 \arrow[r] & 4 \arrow[r] & 2 \arrow[r] & 1
\end{tikzcd}$$

for a total maximal complexity of 47 for complete simple games at $n = 8$. This contrasts with Theorem 3.8 which shows that for arbitrary simple games the complexity can reach 58 at $n = 8$.

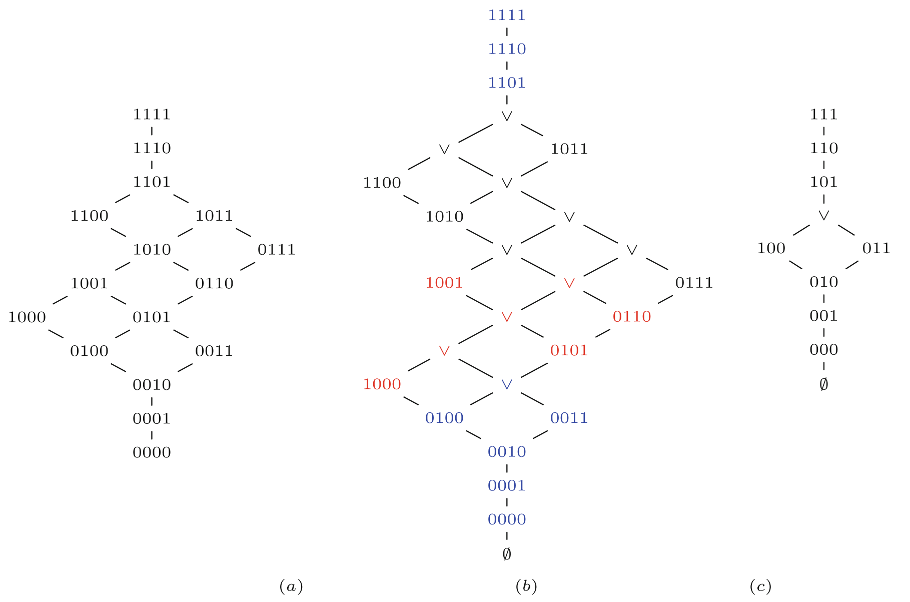

Fig. 5. (a) The majorization lattice E_4 on 4 variables. (b) The 27 complete simple games C_4 on 4 variables. The symbol $\vee$ denotes an element that is join-reducible. Red and blue denote the image under the first map $E_4 \to C_4$ and blue in particular denotes some elements sufficient for the second map $C_4 \to C_3$ to be onto. (c) The 10 complete simple games C_3 on 3 variables. (Color figure online)

References

1. Arora, S., Barak, B., Brunnermeier, M., Ge, R.: Computational complexity and information asymmetry in financial products. Commun. ACM **54**(5), 101–107 (2011)
2. Bolus, S.: Power indices of simple games and vector-weighted majority games by means of binary decision diagrams. Eur. J. Oper. Res. **210**(2), 258–272 (2011)
3. Câmpeanu, C., Ho, W.H.: The maximum state complexity for finite languages. J. Autom. Lang. Comb. **9**(2–3), 189–202 (2004)
4. Champarnaud, J.-M., Pin, J.-E.: A maxmin problem on finite automata. Discrete Appl. Math. **23**(1), 91–96 (1989)
5. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley Series in Computer Science. Addison-Wesley Publishing Co., Reading (1979)

6. Kjos-Hanssen, B.: On the complexity of automatic complexity. Theory Comput. Syst. **61**(4), 1427–1439 (2017)
7. Knuth, D.E.: The Art of Computer Programming. Combinatorial Algorithms, Part 1, vol. 4A. Addison-Wesley, Upper Saddle River (2011)
8. Kurz, S., Tautenhahn, N.: On Dedekind's problem for complete simple games. Int. J. Game Theory **42**(2), 411–437 (2013)
9. Linz, P.: An Introduction to Formal Language and Automata. Jones and Bartlett Publishers Inc., Burlington (2006)
10. Nerode, A.: Linear automaton transformations. Proc. Am. Math. Soc. **9**, 541–544 (1958)
11. Shreve, S.E.: Stochastic Calculus for Finance I: The Binomial Asset Pricing Model. Springer Finance Textbooks. Springer, New York (2004)
12. Sloane, N.J.A.: The online encyclopedia of integer sequences (2018). Sequence A000372