# Chapter 6
# On the Complexity of Conclusive Update

To finitely identify a language means to be able to recognize it with certainty after receiving some (specific) finite sample of the language. Such a finite sample that suffices for finite identification is called *definite finite tell-tale set* (DFTT, for short, see Lange & Zeugmann, 1992; Mukouchi, 1992). One can interpret such a DFTT as the collection of the most characteristic (from a certain point of view) elements of the set. It has also a different connotation that is based on the *eliminative power* of its elements. We can think of the information that is carried by a particular sample of the language in a negative way, as showing which of the hypotheses are inconsistent with the information that has arrived, and thereby eliminating them. A set $S$ is finitely identifiable if its finite subset has the power of eliminating all possibilities under consideration which are different from $S$.

From the characterization of finite identifiability (Mukouchi, 1992), we know that if a class of languages is finitely identifiable, then the identification can be done on the basis of corresponding DFTTs, i.e., finite subsets of the original languages that contain a sample sufficient for finite identifiability. We will call a learner that explicitly uses some DFTTs in the process of identification a *preset learner*. The name derives from the fact that such a learner is equipped with extra information about the DFTTs prior to the identification. A number of issues emerge when analyzing computational properties of the definite finite tell-tales used in identification. Since DFTTs are by no means unique, it can obviously be useful to obtain small definite tell-tales. In this context we distinguish two notions of minimality for DFTTs. A *minimal* DFTT is a DFTT that cannot be further reduced without losing its eliminative power with respect to a class of languages. A *minimal-size* DFTT of a set $S$, is a DFTT that is one of those which are smallest among all possible DFTTs of $S$. In order to investigate the computational complexity of finding such minimal DFTTs, we will have to restrict ourselves to finite classes of languages. Even though it is a very heavy restriction, it creates the possibility of grasping important aspects of the complexity of finite identification. We will next move back to more general cases and investigate how the use of the class of all (minimal) DFTTs can influence the speed of finite

identification.

In the previous chapters we linked the notion of finite identification with the convergence to irrevocable knowledge. The idea of eliminating possibilities that are inconsistent with the incoming data is essentially the same as in the concept of *update* in dynamic epistemic logic (see, e.g., Van Ditmarsch et al., 2007). Presently we discuss, given the epistemic state $S$, the computational complexity of:

1. deciding whether convergence to irrevocable knowledge via update is possible (whether $S$ is finitely identifiable);

2. given that the class is finitely identifiable, finding minimal samples that allow eliminating uncertainty (finding minimal DFTTs);

3. given that the class is finitely identifiable, finding minimal-size samples that allow eliminating uncertainty (finding minimal-size DFTTs).

We argue that the investigations into the complexity of finite identification give a new perspective on the complexity of the emergence of the resulting state, the state of full certainty, that corresponds to the $K$ operator in S5 systems of epistemic logic (see Chapter 2).

The computational tasks can also be interpreted as a motivation for explicitly introducing a new actor, a teacher. Her role is to decide whether learning (given a certain learnability condition) can be successful, and if it can be, to find and provide to the learner minimal samples that will lead to the emergence of knowledge. The analysis of complexity of those tasks assumes a number of conditions on the teacher and the learner: helpfulness of the teacher and eagerness to learn of the learner. Those are not controversial, however they constitute only one of many possible learning and teaching attitudes (other profiles are discussed, in a different setting, in Chapter 7).

The plan of this chapter is as follows. We first recall some relevant learnability notions and the definition and characterization of finite identifiability. We will introduce the notion of *preset learner* that performs identification on the basis of some DFTTs, and characterize the notion using the concept of *subset-driven* learning. Then we will ask the question of how difficult it is to find DFTTs of various kinds. We present the refined notions of *minimal* DFTT, and *minimal-size* DFTT. We show that the problem of finding a minimal-size DFTT is NP-complete, while the problem of finding any minimal DFTT is PTIME computable. Therefore, it can be argued that it is harder for a teacher to provide a minimal-size optimal sample, than just any minimal sufficient information. Then we analyze the possibility of a recursive function that explicitly provides all minimal DFTTs of a finite language. We call the type of finite identification that requires the existence of such a function *strict preset finite identification*. For the case of finite classes of finite languages we apply a computational complexity analysis—here finding the set of all minimal DFTTs turns out to be NP-hard. In the more



general case of infinite classes of finite languages, we also show that there are recursively finitely identifiable classes which are not recursively strict preset finitely identifiable. In the end we compare finite identification with the concept of fastest finite identification and show classes for which recursive finite identifiers exist, but which cannot be recursively finitely identified in the fastest way. That is so because for those classes no recursive function exists that gives access to all minimal DFTTs for each language in the class.

## 6.1 Basic Definitions and Characterization

Let $U \subseteq \mathbb{N}$ be an infinite recursive set, we call any $S \subseteq U$ a language. In the general case, we will be interested in any class of languages that forms an indexed family of recursive languages, i.e., a class $\mathcal{C}$ for which a computable function $f : \mathbb{N} \times U \to \{0, 1\}$ exists that uniformly decides $\mathcal{C}$, i.e.:

$$f(i, w) = \begin{cases} 1 & \text{if } w \in S_i, \\ 0 & \text{if } w \notin S_i. \end{cases}$$

In large parts of this chapter we will also consider $\mathcal{C}$ to be $\{S_1, S_2, \ldots, S_n\}$, a finite class of finite sets, in which case we will use $I_{\mathcal{C}}$ for the set containing indices of sets in $\mathcal{C}$, i.e., $I_{\mathcal{C}} = \{1, \ldots, n\}$.

The notation and basic definition are as introduced in Chapter 2. We recall those that are most important for the content of the present chapter.

Finite identifiability of a class of languages from positive data is defined by the following chain of conditions.

**Definition 6.1.1.** *A learning function $L$:*

1. *finitely identifies $S_i \in \mathcal{C}$ on $\varepsilon$ iff, when inductively given $\varepsilon$, at some point $L$ gives a single output $i$;*

2. *finitely identifies $S_i \in \mathcal{C}$ iff it finitely identifies $S_i$ on every $\varepsilon$ for $S_i$;*

3. *finitely identifies $\mathcal{C}$ iff it finitely identifies every $S_i \in \mathcal{C}$.*

*A class $\mathcal{C}$ is finitely identifiable iff there is a learning function $L$ that finitely identifies $\mathcal{C}$.*

The correspondence between the learning-theoretical setting and the epistemic framework is set to be as in Chapter 4. Namely we take $U := \textsc{Prop}$ an infinite, countable set of propositions, we call any $s \subseteq \textsc{Prop}$ a possible world. A set of possible worlds $S = \{s_1, s_2, \ldots\}$ is an epistemic state. Throughout a large part of this chapter the epistemic states are taken to be finite. Otherwise we assume them to be indexed families of recursive sets of propositions. Accordingly, the

*text (positive presentation)* $\varepsilon$ of $s_i$ is a sound and complete infinite sequence of propositions from PROP allowing repetitions, that are satisfied in $s_i$. For simplicity we will continue here with the number-theoretical framework, but we would like the reader to bear in mind that the epistemic interpretation of these results is straightforward.

Let us recall the necessary and sufficient condition for finite identifiability (Lange & Zeugmann, 1992; Mukouchi, 1992). It involves a modified, stronger notion of finite tell-tale (Angluin, 1980), namely the *definite finite tell-tale set*.

**Definition 6.1.2** (Mukouchi 1992). *A set $D_i$ is a definite finite tell-tale set for $S_i \in \mathcal{C}$ if*

1. $D_i \subseteq S_i$,
2. $D_i$ *is finite, and*
3. *for any index $j$, if $D_i \subseteq S_j$ then $S_i = S_j$.*

Finite identifiability can be then characterized in the following way.

**Theorem 6.1.3** (Mukouchi 1992). *A class $\mathcal{C}$ is finitely identifiable from positive data iff there is an effective procedure $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$, given by $n \mapsto \mathcal{D}_n$, that on input $i$ produces a definite finite tell-tale of $S_i$.*

In other words, each set in a finitely identifiable class contains a finite subset that distinguishes it from all other sets in the class. Moreover, for the effective identification it is required that there is a *recursive* procedure that provides such DFTTs.

Let us first observe that if a language $S_i$ contains a DFTT, then every text for $S_i$ enumerates all elements of this DFTT in finite time.

**Proposition 6.1.4.** *If $\varepsilon$ is a text for $S_i \in \mathcal{C}$ and $S$ is a finite subset of $S_i$ (in particular a DFTT of $S_i$), then there is an $n \in \mathbb{N}$, such that* $\mathrm{set}(\varepsilon\lceil n)$ *is a superset of $S$.*

*Proof.* Let us take a finite $S \subseteq S_i$, and $\varepsilon$—a text for $S_i$. Assume for contradiction there is no $n \in \mathbb{N}$ such that $\mathrm{set}(\varepsilon\lceil n)$ is a superset of $S$. Then that means that there is $k \in \mathbb{N}$ such that $k \in S \subseteq S_i$ and for all $n$, $\varepsilon_n \neq k$. This contradicts the definition of text. □

Theorem 6.1.3 gives the characterization of finite identification in terms of a recursive procedure that *generates* DFTTs. Below we present a new way of tell-tale sets being given by a decision procedure. We will call such a procedure a *dftt-function*.

**Definition 6.1.5.** *Let $\mathcal{C}$ be an indexed family of recursive sets. The* dftt-function *for $\mathcal{C}$ is a recursive function $f_{dftt} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$, such that:*



1. *if $f_{dftt}(S,i) = 1$, then $S$ is a DFTT of $S_i$;*
2. *for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, such that $f_{dftt}(S,i) = 1$.*

A first observation about the dftt-function is that it cannot attribute two $i, j \in \mathbb{N}$, such that $i \neq j$ to one finite set $S$.

**Proposition 6.1.6.** *Let $\mathcal{C}$ be a class of languages and $f_{dftt}$ be a dftt-function for $\mathcal{C}$. Then there is no finite $S \subseteq \mathbb{N}$ such that for some $i, j \in \mathbb{N}$, such that $i \neq j$ and $f_{dftt}(S,i) = 1$ and $f_{dftt}(S,j) = 1$.*

*Proof.* Assume that there is a finite $S \subseteq \mathbb{N}$ and $i, j \in \mathbb{N}$ such that $f_{\mathrm{dftt}}(S,j) = 1$ and $f_{\mathrm{dftt}}(S,i) = 1$. Then, by definition of $f_{\mathrm{dftt}}$, $S$ is a DFTT of both $S_i$ and $S_j$. By the definition of DFTT, $i = j$. □

Now we will show that in fact the condition given in Theorem 6.1.3 is equivalent to the existence of such dftt-function.

**Theorem 6.1.7.** *A class $\mathcal{C}$ is finitely identifiable from positive data iff there is a dftt-function for $\mathcal{C}$.*

*Proof.* ($\Rightarrow$) Let us assume that $\mathcal{C}$ is finitely identifiable. Then, by Theorem 6.1.3 there is an effective procedure $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$, given by $n \mapsto \mathcal{D}_n$, that on input $i$ produces all elements of a definite finite tell-tale of $S_i$. Let $S \subseteq \mathbb{N}$ be a finite set and $i \in \mathbb{N}$. We define $f : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ in the following way:

$$f(S,i) = \begin{cases} 1 & \text{if } \mathcal{D}_i = S; \\ 0 & \textit{otherwise.} \end{cases}$$

Let us observe that $f$ is a dftt-function for $\mathcal{C}$:

1. $f$ is recursive: given $S$ and $i$ the function $f$ uses $\mathcal{D}$ to produce a DFTT of $S_i$, and then compares the obtained $\mathcal{D}_i$ with $S$. Such a $\mathcal{D}_i$ always exists by the definition of $\mathcal{D}$;
2. if $f(S,i) = 1$ then $S$ is obviously a DFTT of $S_i$;
3. for all $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$ such that $f(S,i) = 1$, by the definition of $\mathcal{D}$.

($\Leftarrow$) Let us take a class $\mathcal{C}$ and assume that there is a dftt-function for $\mathcal{C}$. Let us take $S_i \in \mathcal{C}$. The standard text $\varepsilon^{\mathrm{st}}$ for $S_i$ is defined in the following way:

$$\varepsilon^{\mathrm{st}}_0 = \mu n(n \in S_i), \text{ and}$$

$$\varepsilon^{\mathrm{st}}_n = \begin{cases} n & \text{if } n \in S_i; \\ \varepsilon^{\mathrm{st}}_{n-1} & \text{otherwise.} \end{cases}$$

We define $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$ in the following way: On input $i$, $\mathcal{D}$ constructs the standard text for $S_i$ in a step by step manner. At each step $n$, $\mathcal{D}$ performs a search for a $S \subseteq \text{set}(\varepsilon^{\text{st}}\restriction n)$ such that $f_{\text{dftt}}(S, i) = 1$. The first one found in this manner is taken to be $\mathcal{D}_i$. By Definition of $f_{\text{dftt}}$ and Proposition 6.1.4, $\mathcal{D}$ is recursive and $\mathcal{D}_i$ is a DFTT of $S_i$. □

We have shown that DFTTs for a finitely identifiable class can be given in two equivalent ways. It is important to remember that $f_{\text{dftt}}$ may not recognize all DFTTs of a given language, but it is guaranteed to 'know about' at least one.

## 6.2 Preset Learning

Let us now turn to our central notion of *preset learning*. Intuitively, a preset learning function uses a recursive decision function, such as the dftt-function defined in the previous section, as a guide in the process of identification.

**Definition 6.2.1.** *Let $\mathcal{C} = \{S_i \mid i \in \mathbb{N}\}$ be a class of languages, $\varepsilon$—a text for some $S_i \in \mathcal{C}$, and $f : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0, 1\}$ be a recursive function. A* preset learning function $L$ based on $f$ *is defined in the following way:*

$$L(\varepsilon\restriction n) = \begin{cases} \mu j \ \text{set}(\varepsilon\restriction n) \subseteq S_j & \text{if for that } j\ \exists S \subseteq \text{set}(\varepsilon\restriction n)\ f(S, j) = 1 \\ & \ \& \ \forall k < n\ L(\varepsilon\restriction k) = \uparrow; \\ \uparrow & \text{otherwise.} \end{cases}$$

It is easy to see that on any text for a language in the class such a function has at most one integer value. In general we will call such learning functions (at most) once defined.

**Definition 6.2.2.** *A learning function $L$ is* (at most) once defined *on $\mathcal{C}$ iff for any text $\varepsilon$ for a language from $\mathcal{C}$ and $n, k \in \mathbb{N}$ such that $n \neq k$: $L(\varepsilon\restriction n) = \uparrow$ or $L(\varepsilon\restriction k) = \uparrow$.*

**Proposition 6.2.3.** *Every preset learning function is (at most) once defined.*

*Proof.* Let $\mathcal{C}$ be a class of languages. Assume that $L$ is a preset learning function based on recursive $f$, and, for contradiction, that $L$ is not (at most) once defined on $\mathcal{C}$. Then, there is $\varepsilon$—a text for some $S_i \in \mathcal{C}$ and $\ell, n \in \mathbb{N}$ such that $\ell \neq n$, $L(\varepsilon\restriction \ell) \neq \uparrow$ and $L(\varepsilon\restriction n) \neq \uparrow$. Assume that $\ell < n$. Since $L$ is total, there is an $i$ such that $L(\varepsilon\restriction n) = i$, and so, by the definition of $L$, $\exists S \subseteq \text{set}(\varepsilon\restriction n)\ f(S, i) = 1$ and $\forall k < n\ L(\varepsilon\restriction k) = \uparrow$. The latter gives a contradiction with the assumption that $L(\varepsilon\restriction \ell) \neq \uparrow$. □

Moreover, we can show that if for every $i \in \mathbb{N}$, $f$ judges at least one finite $S \subseteq S_i$ positively then the preset learning function based on $f$ is recursive.



**Proposition 6.2.4.** *Let $L$ be a preset learning function based on $f$. If $f$ satisfies Condition 2 of Definition 6.1.5, i.e., for all $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$ such that $f(S, i) = 1$, then $L$ is recursive on any finitely identifiable class.*

We will now show that preset learners can identify every finitely identifiable class.

**Proposition 6.2.5.** *If a class $\mathcal{C}$ is finitely identifiable then it is finitely identified by a preset learner.*

*Proof.* Assume that $\mathcal{C}$ is finitely identifiable, then by the Theorem 6.1.7, there is a dftt-function $f_{\text{dftt}}$ for $\mathcal{C}$. We will show that $L$, the recursive preset learner based on $f_{\text{dftt}}$ finitely identifies $\mathcal{C}$. First, let us observe that by Proposition 6.2.4, $L$ is recursive. By Proposition 6.2.3 we have that for any $\varepsilon$ text for some $S_i \in C$, $L$ is (at most) once defined. Let us take $\varepsilon$ a text for $S_i \in \mathcal{C}$. By the definition of text and DFTT for all $j \neq i$, there is no $n \in \mathbb{N}$ and no $S$ such that $S \subseteq \text{set}(\varepsilon\restriction n)$ & $f(S, j) = 1$. By the definition of $f_{\text{dftt}}$ we get that $\exists S \subseteq S_i\ f(S, i) = 1$, and by Proposition 6.1.4 there $\exists n, S$ $(S \subseteq \text{set}(\varepsilon\restriction n)$ & $f(S, i) = 1)$. Take the smallest such $n$. Then $L(\varepsilon\restriction n) = i$. □

We have shown that a preset learning function based on a dftt-function can identify any finitely identifiable class. In the next section we will discuss some further properties of preset learning.

**Set-Drivenness and Subset-Drivenness** The fact that the preset learner based on a dftt-function is universal with respect to finite identification indicates that for finite learning it is enough to care at each step only about the content of the finite sequence presented so far. In particular, a preset learner only checks whether the sequence includes a subset with certain properties. It does not pay attention to the order of elements and repetitions. Learning functions that work this way are called set-driven.

**Definition 6.2.6** (Wexler & Cullicover 1980)**.** *Let $\mathcal{C}$ be an indexed family of recursive sets. A learning function $L$ is said to be* set-driven *with respect to $\mathcal{C}$ iff for any two texts $\varepsilon_1$ and $\varepsilon_2$ for some languages in $\mathcal{C}$ and any two $n, k \in \mathbb{N}$, if $\text{set}(\varepsilon_1\restriction n) = \text{set}(\varepsilon_2\restriction k)$, $L(\varepsilon_1\restriction n) \neq \uparrow$ and $L(\varepsilon_2\restriction k) \neq \uparrow$ (i.e., they both have a natural number value), then $L(\varepsilon_1\restriction n) = L(\varepsilon_2\restriction k)$.*

It has been shown that set-drivenness does not restrict the power of finite identification.[1] This is different from the general case of identification in the limit, where set-drivenness does restrict the power of identification.

[1] In their proof of Theorem 6.2.7, Lange & Zeugmann construct a learner very similar to our preset learning function. We would like to thank the anonymous reviewer of *The 23rd Annual Conference of Learning Theory 2010* for pointing us in this direction.

**Theorem 6.2.7** (Lange & Zeugmann 1996). *A class $\mathcal{C}$ is finitely identifiable if and only if $\mathcal{C}$ is finitely identified by a set-driven learner.*

We will show that any preset learning function based on a dftt-function is set-driven.

**Theorem 6.2.8.** *Let $\mathcal{C}$ be a class of languages, and $f_{dftt}$ be a dftt-function for $\mathcal{C}$. If $L$ is a preset learning function based on $f_{dftt}$, then $L$ is set-driven with respect to $\mathcal{C}$.*

*Proof.* Take $\mathcal{C}$, $f_{\text{dftt}}$ and $L$ as specified in the theorem. Assume that $\varepsilon_1$ and $\varepsilon_2$ are texts for some languages in $\mathcal{C}$; $n, k \in \mathbb{N}$; $\text{set}(\varepsilon_1{\restriction}n) = \text{set}(\varepsilon_2{\restriction}k)$; $L(\varepsilon_1{\restriction}n) \neq \uparrow$ and $L(\varepsilon_2{\restriction}k) \neq \uparrow$ (they both have an integer value). Assume that $L(\varepsilon_1{\restriction}n) = i$. Then, by the definition of $L$, $\text{set}(\varepsilon_1{\restriction}n) \subseteq S_i$, $\exists S \subseteq \text{set}(\varepsilon_1{\restriction}n)\ f(S,i) = 1$ and $\forall \ell < n\ L(\varepsilon_1{\restriction}\ell) = \uparrow$. The same holds for $L(\varepsilon_2{\restriction}k)$ and some $j \in \mathbb{N}$. We have to show that $i = j$.

Assume for contradiction that $i \neq j$. Then, since $\text{set}(\varepsilon_1{\restriction}n) = \text{set}(\varepsilon_2{\restriction}k)$, $\exists S \subseteq \text{set}(\varepsilon_1{\restriction}n) \subseteq \text{set}(\varepsilon_2{\restriction}k)\ f(S,i) = 1$ and $\text{set}(\varepsilon_2{\restriction}k) \subseteq S_j$. This means that there is a finite set $S$ that is a DFTT for $S_i$ and at the same time $S \subseteq S_j$ for some $j \neq i$. This gives a contradiction with the definition of DFTT. □

A stronger notion of set-drivenness is possible. Definition 6.2.6 restricts the condition to those situations in which $L$ gives an integer value. The alternative concept is as follows.

**Definition 6.2.9** (Wexler & Cullicover 1980). *Let $\mathcal{C}$ be an indexed family of recursive sets. A learning function $L$ is said to be* strongly set-driven *with respect to $\mathcal{C}$ iff for any two texts $\varepsilon_1$ and $\varepsilon_2$ for some languages in $\mathcal{C}$ and any two $n, k \in \mathbb{N}$, if $\text{set}(\varepsilon_1{\restriction}n) = \text{set}(\varepsilon_2{\restriction}k)$, then $L(\varepsilon_1{\restriction}n) = L(\varepsilon_2{\restriction}k)$.*

The preset learner based on an $f_{\text{dftt}}$ for $\mathcal{C}$ is not strongly set-driven with respect to $\mathcal{C}$. Consider the folowing simple example. Let $\mathcal{C} = \{S_1 = \{1,2,4\}, S_2 = \{1,3,4\}\}$, and let $\varepsilon_1 = \langle 1,2,4,\ldots\rangle$ and $\varepsilon_2 = \langle 1,4,2\ldots\rangle$ be two texts for $S_1$. Let us compare the outputs of the learning function in the two cases of the initial segments of $\varepsilon_1$ and $\varepsilon_2$: $L(\langle 1,2,4\rangle) = \uparrow$ and $L(\langle 1,4,2\rangle) = 1$. The content of the two sequences is the same but the outputs of $L$ are different.

From Theorem 6.2.7 we know that set-drivenness does not restrict finite identifiability. The notion of preset learning leads to a concept of subset-driven learning, that is itself related to set-driven learning.[2]

**Definition 6.2.10.** *Let $\mathcal{C}$ be an indexed family of recursive sets. A learning function $L$ is* subset-driven *with respect to $\mathcal{C}$ iff for any two texts $\varepsilon_1$ and $\varepsilon_2$ for some languages in $\mathcal{C}$, and any $n, k \in \mathbb{N}$:*

[2] In fact, if one considers (instead of once-defined functions) functions that keep outputting the same value after having given a value once, then the concepts of strongly set-driven and subset-driven coincide. This would also make the anomaly vanish that preset learners are not strongly set-driven.



- *If $L(\varepsilon_1{\restriction}n) = \downarrow$ and $\text{set}(\varepsilon_1{\restriction}n) \subseteq \text{set}(\varepsilon_2{\restriction}k)$ and for all $\ell < k$, $L(\varepsilon_2{\restriction}\ell) = \uparrow$, then $L(\varepsilon_1{\restriction}n) = L(\varepsilon_2{\restriction}k)$.*

In other words, assume that a subset-driven learning function on an initial segment $\varepsilon{\restriction}n$ gives an integer answer. Then if there is some other text that at some point enumerates all elements of $\varepsilon{\restriction}n$, and up to that point no answer was given, then the function is bound to give the same integer answer.

**Theorem 6.2.11.** *Let $\mathcal{C}$ be a class of languages, and $f_{dftt}$ be a dftt-function for $\mathcal{C}$. If $L$ is a preset learning function based on $f_{dftt}$, then $L$ is subset-driven with respect to $\mathcal{C}$.*

*Proof.* Take $\mathcal{C}$, $f_{\text{dftt}}$ and $L$ as specified in the theorem. Assume that $\varepsilon_1$ and $\varepsilon_2$ are texts for some languages in $\mathcal{C}$; $L(\varepsilon_1{\restriction}n) = \downarrow$ and $\text{set}(\varepsilon_1{\restriction}n) \subseteq \text{set}(\varepsilon_2{\restriction}k)$ and for all $\ell < k$, $L(\varepsilon_2{\restriction}\ell) = \uparrow$. We have to show that then $L(\varepsilon_1{\restriction}n) = L(\varepsilon_2{\restriction}k)$.

By the fact that $L(\varepsilon_1{\restriction}n) = \downarrow$, we know that there is $i \in \mathbb{N}$ such that $\text{set}(\varepsilon_1{\restriction}n) \subseteq S_i$ and that $\exists S \subseteq \text{set}(\varepsilon_1{\restriction}n)\ f(S,i) = 1$. Then $\exists S \subseteq \text{set}(\varepsilon_1{\restriction}n) \subseteq \text{set}(\varepsilon_2{\restriction}k)$ such that $f(S,i) = 1$, i.e., $\text{set}(\varepsilon_2{\restriction}k)$ includes a finite set $S$ such that $S$ is a DFTT for $S_i$. Hence, by the definitions of text and DFTT, $\text{set}(\varepsilon_2{\restriction}k) \subseteq S_i$. Moreover, one of the assumptions is that for all $\ell < k$, $L(\varepsilon_2{\restriction}\ell) = \uparrow$. Therefore, $L(\varepsilon_2{\restriction}\ell) = i$. □

The connection between subset-driven finite identifiers and preset learners is even tighter. Every subset-driven learning function that finitely identifies a class is a preset learner (with respect to some $f$).

**Theorem 6.2.12.** *Assume $\mathcal{C}$ is a class of languages and $\mathcal{C}$ is finitely identified by a subset-driven learning function $L$. Then $L$ is a preset learner (with respect to some $f$).*

*Proof.* Let subset-driven learning function $L$ finitely identify $\mathcal{C}$. We define $f$ in the following way:

$$f(s,i) = 1 \text{ iff, for some } T \subseteq S,\ L(\hat{T}) = i.$$

We show that $L_f$, preset learner with respect to $f$, is equal to $L$.

Let us take $\varepsilon$ a text for some language in $\mathcal{C}$ and take $n$ such that for all $k < n$, $L(\varepsilon{\restriction}k) = \uparrow$. It is sufficient to show that in this case $L(\varepsilon{\restriction}n) = L_f(\varepsilon{\restriction}n)$.

First, assume that $L(\varepsilon{\restriction}n) = \uparrow$. Then, since $L$ is subset-driven, for no $S \subseteq \text{set}(\varepsilon{\restriction}n)$, $L(\hat{S}) \neq \uparrow$ (otherwise $L(\varepsilon{\restriction}n)$ would have the same value). So for all $S \subseteq \text{set}(\varepsilon{\restriction}n)$ and all $i$, $f(S,i) = 0$. Hence, $L_f(\varepsilon{\restriction}n) = \uparrow$.

Next, assume that $L(\varepsilon{\restriction}n) = i$. Then, because of the set-drivenness of $L$, $f(\text{set}(\varepsilon{\restriction}n), i) = 1$ and $L_f(\varepsilon{\restriction}n) = i$ immediately follows. □

Having established the set- and subset-drivenness of preset finite identifiers, we will now turn to investigating the complexity of finding DFTTs that govern the preset finite identification. Until now we have focused on the availability of any DFTT for each language from a class. Of course, a language can have many different DFTTs. In the next section we will distinguish different types of DFTT and discuss their usefulness for finite identification.

## 6.3 Eliminative Power and Complexity

Identifiability in the limit (Gold, 1967) of a class of languages guarantees the existence of a reliable strategy that allows for convergence to a correct hypothesis for every language from the class. The exact moment at which a correct hypothesis has been reached is not known and in general can be uncomputable. Things are different for finite identifiability. Here, the learning function is allowed to answer only once. Hence, the conjecture is based on certainty. In other words, the learner must know that the answer she gives is true, because there is no opportunity for a change of mind later.

Knowing that one hypothesis is true means being able to exclude all other possibilities. In this section we define the notion of *eliminative power* of a piece of information, which reflects the informative strength of data with respect to a certain class of sets.

**Definition 6.3.1.** *Let us take $\mathcal{C}$ an indexed class of recursive languages, and $x \in \bigcup \mathcal{C}$.* The eliminative power of $x$ with respect to $\mathcal{C}$ *is determined by a function $El_{\mathcal{C}} : \bigcup \mathcal{C} \to \mathcal{P}(\mathbb{N})$, such that:*

$$El_{\mathcal{C}}(x) = \{i \mid x \notin S_i \ \&\ S_i \in \mathcal{C}\}.$$

*Additionally, we will write $El_{\mathcal{C}}(X)$ for $\bigcup_{x \in X} El_{\mathcal{C}}(x)$.*

In other words, function $El_{\mathcal{C}}$ takes $x$ and outputs the set of indices of all the sets in $\mathcal{C}$ that are inconsistent with $x$, and therefore in the light of $x$ they can be "eliminated". We can now characterize finite identifiability in terms of the eliminative power.

**Proposition 6.3.2.** *A set $D_i$ is a definite tell-tale set of $S_i \in \mathcal{C}$ iff*

1. *$D_i \subseteq S_i$,*
2. *$D_i$ is finite, and*
3. *$El_{\mathcal{C}}(D_i) = \mathbb{N} - \{i\}$.*

Moreover, from Theorem 6.1.3 we know that finite identifiability of an indexed family of recursive languages requires that every set in a class has a DFTT. Formally:

**Theorem 6.3.3.** *A class $\mathcal{C}$ is finitely identifiable from positive data iff there is an effective procedure $\mathcal{D} : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$, given by $n \mapsto \mathcal{D}_n$, that on input $i$ produces a finite set $D_i \subseteq S_i$, such that*

$$El_{\mathcal{C}}(D_i) = \mathbb{N} - \{i\}.$$



### 6.3.1 The Complexity of Finite Identifiability Checking

As has already been mentioned in the introduction to this chapter, we aim to analyze the computational complexity of finding DFTTs. In order to do that we restrict ourselves to finite classes of finite sets. One may ask about the purpose of further reduction of sets that are already finite. In fact, if a finite class of finite sets is finitely identifiable, then each element of the class is already its own DFTT. However, finite sets can be much larger than their DFTTs. For example, we can take a class of the following form:

$$\mathcal{C} = \{S_i = \{2i, 2^i \text{ first odd natural numbers}\} \mid i = 1, \ldots, n\}.$$

In this case reduction to the minimal information that suffices for finite identification, $2i$, makes a significant difference in the complexity of the process of learning. The learner can simply disregard all odd numbers and wait for an even number.

**Theorem 6.3.4.** *Checking whether a finite class of finite sets is finitely identifiable is polynomial with respect to the number of sets in the class and the maximal cardinality of sets in the class.*

*Proof.* The procedure consists of computing $El_{\mathcal{C}}(x)$ and checking whether for each $S_i \in \mathcal{C}$, $El(S_i) = I_{\mathcal{C}} - \{i\}$, where $I_{\mathcal{C}} = \{i \mid S_i \in \mathcal{C}\}$.

Let us first focus on computing $El_{\mathcal{C}}(x)$ for $x \in \bigcup \mathcal{C}$. We take a class $\mathcal{C}$ and assume that $|\mathcal{C}| = m$, and that the largest set in $\mathcal{C}$ has $n$ elements.

In the first steps we have to obtain $\bigcup \mathcal{C}$. After this, we list for each element of $\bigcup \mathcal{C}$ the indices of the sets to which the element does not belong. In this step we have computed $El_{\mathcal{C}}(x)$ for each $x \in \bigcup \mathcal{C}$. All components of this procedure can clearly be performed in polynomial time with respect to $m$ and $n$. It remains to check whether for all $S_i \in \mathcal{C}$, $\bigcup_{x \in S_i} El_{\mathcal{C}}(x) = I_{\mathcal{C}} - \{i\}$. This involves essentially only the operation of sum. □

From this analysis we conclude that checking whether a finite class of finite sets is finitely identifiable is a quite easy, polynomial task. Nevertheless, as we saw in the example in the beginning of this section, it can be time consuming if $n$ and $m$ are large numbers.

### 6.3.2 Minimal Definite Finite Tell-Tale

We are now ready to introduce one of the two nonequivalent notions of minimality of the DFTTs. We will call $D_i$ a minimal DFTT of $S_i$ in $\mathcal{C}$ if and only if all the elements of the sets in $D_i$ are essential for finite identification of $S_i$ in $\mathcal{C}$, i.e., taking an element out of the set $D_i$ will decrease its eliminative power with respect to $\mathcal{C}$, and hence it will no longer be a DFTT. We will observe that a language can have many minimal DFTTs of different cardinalities. This will give us a cause to introduce another notion of minimality—minimal-size DFTT.

Learning functions are bound to be guided by the elements that are presented to them in texts. In order to converge quickly there is no reason for the learner to look especially for a *certain* minimal or minimal-size DFTT, because those might not appear soon enough in the text. However, being able to recognize *all* minimal DFTTs can intuitively guarantee that the right answer occurs as soon as it is objectively possible. If it is not the time of convergence but the memory of the learner that we want to spare, having access to all minimal-size DFTTs is obviously useful. Finding the minimal-size DFTTs can certainly be attributed to an efficient teacher, who looks for an optimal sample that allows identification.

**Definition 6.3.5.** *Let us take a finitely identifiable indexed family of recursive languages $\mathcal{C}$, and $S_i \in \mathcal{C}$. A* minimal DFTT *of $S_i$ in $\mathcal{C}$ is a $D_i \subseteq S_i$, such that*

1. *$D_i$ is a DFTT for $S_i$ in $\mathcal{C}$, and*
2. *$\forall X \subset D_i \; El_{\mathcal{C}}(X) \neq I_{\mathcal{C}} - \{i\}$.*

**Theorem 6.3.6.** *Let $\mathcal{C}$ be a finitely identifiable finite class of finite sets. Finding a minimal DFTT of $S_i \in \mathcal{C}$ can be done in polynomial time.*

*Proof.* Assume that the class $\mathcal{C}$:

1. is finitely identifiable;
2. is finite;
3. consists only of finite sets.

From the assumptions 1 and 3, we know that for each $S_i \in \mathcal{C}$ a DFTT exists, in fact $S_i$ is its own DFTT.

The following procedure yields a minimal DFTT for each $S_i \in \mathcal{C}$.

We want to find a set $X \subseteq S_i$ such that

$$El(X) = I_{\mathcal{C}} - \{i\}, \text{ but } \forall Y \subset X \; El(Y) \neq I_{\mathcal{C}} - \{i\}.$$

First we set $X := S_i$.

We look for the minimal $x \in X$ such that $El(X - \{x\}) = I_{\mathcal{C}} - \{i\}$. If there is no such element, $X$ is the desired DFTT. If there is such an $x$, we set $X := X - \{x\}$, and repeat the procedure.

Let $|S_i| = n$, where $|\cdot|$ stands for cardinality. The number of comparisons needed for finding a minimal DFTT of $S_i$ in $\mathcal{C}$ is bounded by $n^2$. □

**Example 6.3.7.** *Let us consider the class*

$$\mathcal{C} = \{S_1 = \{1,3,4\}, S_2 = \{2,4,5\}, S_3 = \{1,3,5\}, S_4 = \{4,6\}\}.$$

*The procedure of finding minimal DFTTs for sets in $\mathcal{C}$ is as follows.*



| $x$ | $El_{\mathcal{C}}(x)$ |
|---|---|
| 1 | $\{2,4\}$ |
| 2 | $\{1,3,4\}$ |
| 3 | $\{2,4\}$ |
| 4 | $\{3\}$ |
| 5 | $\{1,4\}$ |
| 6 | $\{1,2,3\}$ |

Table 6.1: Eliminative power of the elements in $\bigcup\mathcal{C}$ with respect to $\mathcal{C}$

| sct | a minimal DFTT |
|---|---|
| $\{1,3,4\}$ | $\{3,4\}$ |
| $\{2,4,5\}$ | $\{4,5\}$ |
| $\{1,3,5\}$ | $\{3,5\}$ |
| $\{4,6\}$ | $\{6\}$ |

Table 6.2: DFTTs of $\mathcal{C}$

1. *We construct a list of the elements from $\bigcup\mathcal{C}$.*
2. *With each element $x$ from $\bigcup\mathcal{C}$ we associate $El_{\mathcal{C}}(x) = \{i \mid x \notin S_i\}$, i.e., the set of indices of sets to which $x$ does not belong (names of sets that are inconsistent with $x$). Table 6.1 shows the result of the two first steps for $\mathcal{C}$.*
3. *The next step is to find minimal DFTTs for every set in the class $\mathcal{C}$. As an example, let us take the first set $S_1 = \{1,2,3\}$. We order elements of $S_1$, and take the first element of the ordering. Let it be 1. We compute $El_{\mathcal{C}}(S_1 - \{1\})$, it turns out to be $\{2,3,4\}$. We therefore accept the set $\{3,4\}$ as a smaller DFTT for $S_1$. Then we try to further reduce the obtained DFTT, by checking the next element in the ordering, let it be 3. $El_{\mathcal{C}}(\{3,4\} - \{3\}) = \{4\} \neq \{2,3,4\}$, so 3 cannot be subtracted without loss of eliminative power. We perform the same check for the last singleton $\{4\}$. It turns out that $\{3,4\}$ cannot further be reduced. We give $\{3,4\}$ as a minimal DFTT of $S_1$.*[3]
4. *We perform the same procedure for all the sets in $\mathcal{C}$. As a result we get minimal DFTTs for each $S_i \in \mathcal{C}$ presented in Table 6.2.*

[3] Checking only singletons is enough because the eliminative power of sets is defined as the sum of the eliminative power of its elements.

### 6.3.3 Minimal-Size Definite Finite Tell-Tale

Minimal DFTTs of a language include all information that is enough to exclude other possibilities and involve no redundant data. We can use the notion of eliminative power to construct a procedure for finding minimal-size DFTTs of a finitely identifiable finite class of finite sets $\mathcal{C}$. Minimal-size DFTTs are the minimal DFTTs of smallest cardinality.

We assume that $|\mathcal{C}| = m$. To find a DFTT of minimal size for the set $S_i \in \mathcal{C}$, one has to perform a search through all the subsets of $S_i$ starting from singletons, looking for the first $X_i$, such that $El(X_i) = I_{\mathcal{C}} - \{i\}$.

DFTTs of minimal size need not be unique. Which one is encountered first depends on the manner of performing the search. Below we describe a possible way of searching for minimal-size DFTTs on the example discussed before.

**Example 6.3.8.** *Let us consider again the class from Example 6.3.7, namely*

$$\mathcal{C} = \{S_1 = \{1,3,4\}, S_2 = \{2,4,5\}, S_3 = \{1,3,5\}, S_4 = \{4,6\}\}.$$

1. *We construct a list of the elements from* $\bigcup\mathcal{C}$.

2. *With each element* $x$ *from* $\bigcup\mathcal{C}$ *we associate* $El_{\mathcal{C}}(x) = \{i \mid x \notin S_i\}$, *i.e., the set of hypotheses for sets to which* $x$ *does not belong (names of sets that are inconsistent with* $x$*). Table* 6.1 *presents the result of the two first steps for* $\mathcal{C}$.

3. *The next step is to find minimal-size DFTTs for every set in the class* $\mathcal{C}$. *As an example, let us take the first set* $S_1 = \{1,3,4\}$. *We are looking for* $X \subseteq S_1$ *of minimal size, such that* $El_{\mathcal{C}}(X) = I_{\mathcal{C}} - \{1\}$.

   (a) *We look for* $\{x\}$ *such that* $x \in S_1$ *and* $El_{\mathcal{C}}(\{x\}) = \{2,3,4\}$. *There is no such singleton.*

   (b) *We look for* $\{x,y\}$ *such that* $x, y \in S_1$ *and* $El_{\mathcal{C}}(\{x\}) = \{2,3,4\}$. *There are two such sets:* $\{1,4\}$ *and* $\{3,4\}$.

4. *We perform the same procedure for all* $S_i \in \mathcal{C}$. *As a result we get minimal-size DFTTs for each of* $\mathcal{C}$, *the result is presented in Table 6.3.*

Let us now compare the two resulting reductions of sets from $\mathcal{C}$ (see Table 6.4). The case of $S_2$ shows that the two procedures give different outcomes.

**Running time** Let us now analyze and discuss the running time of this procedure. First we need to compute $El_{\mathcal{C}}(x)$ for $\bigcup\mathcal{C}$. From the Theorem 6.3.4 we know that it can be done in polynomial time. Now, let us approximate the number of steps needed to find a minimal-size DFTT of a chosen set $S_i \in \mathcal{C}$. We again assume that $|\mathcal{C}| = m$, and $S_i$ has $n$ elements. In the procedure described above we



| set | minimal-size DFTTs |
|---|---|
| $\{1,3,4\}$ | $\{1,4\}$ or $\{3,4\}$ |
| $\{2,4,5\}$ | $\{2\}$ |
| $\{1,3,5\}$ | $\{1,5\}$ or $\{3,5\}$ |
| $\{4,6\}$ | $\{6\}$ |

Table 6.3: Minimal-size DFTTs of $\mathcal{C}$

| set | a minimal DFTT | minimal-size DFTTs |
|---|---|---|
| $\{1,3,4\}$ | $\{3,4\}$ | $\{1,4\}$ or $\{3,4\}$ |
| $\{2,4,5\}$ | $\{4,5\}$ | $\{2\}$ |
| $\{1,3,5\}$ | $\{3,5\}$ | $\{1,5\}$ or $\{3,5\}$ |
| $\{4,6\}$ | $\{6\}$ | $\{6\}$ |

Table 6.4: A comparison of minimal and minimal-size DFTTs of $\mathcal{C}$

performed a search through, in the worst case, all combinations from 1 to $|S_i|$, to find the right set $X \subseteq S_i$, such that $El_{\mathcal{C}}(X)$ satisfies the condition of eliminating all hypothesis but $h_i$. So, for each set $S_i$, the number of comparisons that have to be performed is:

$$n + \frac{n!}{2!(n-2)!} + \frac{n!}{3!(n-3)!} + \ldots + 1 = 2^{n-1}$$

**Computational Complexity** It is costly to find minimal-size DFTTs. As we have seen above, our procedure leads to a complete search through the large space of all subsets of a language. We call this computational problem the MINIMAL-SIZE DFTT Problem, and define it formally below. In words, the problem can be phrased as checking whether $S_i \in \mathcal{C}$ has a DFTT of size $k$ or smaller.

**Definition 6.3.9** (MINIMAL-SIZE DFTT Problem).

**Instance** *A finite class of finite sets* $\mathcal{C}$, *a set* $S_i \in \mathcal{C}$, *and a positive integer* $k \leq |S_i|$.

**Question** *Is there a minimal DFTT* $X_i \subseteq S_i$ *of size* $\leq k$?

We are going to show that the MINIMAL-SIZE DFTT Problem is NP-complete by pointing out that it is equivalent to the MINIMUM COVER Problem, which is know to be NP-complete (Karp, 1972). Let us recall it below.

**Definition 6.3.10** (MINIMAL COVER Problem).

**Instance:** *Collection $P$ of subsets of a finite set $F$, positive integer $k \leq |P|$.*

**Question:** *Does $P$ contain a cover for $X$ of size $k$ or less, i.e., a subset $P' \subseteq P$ with $|P'| \leq k$ such that every element of $X$ belongs to at least one member of $P'$?*

**Theorem 6.3.11.** *The* MINIMAL-SIZE DFTT *Problem is NP-complete.*

*Proof.* First, let us observe that by Theorem 6.3.3, MINIMAL-SIZE DFTT Problem is equivalent to the following Problem:

**Definition 6.3.12** (MINIMAL-SIZE DFTT′ Problem).

**Instance:** *Collection $\{El(x) \mid x \in S_i\}$, positive integer $k \leq |S_i|$.*

**Question:** *Does $\{El(x) \mid x \in S_i\}$ contain a cover for $I_C - \{i\}$ of size $k$ or less, i.e., a subset $Y_i \subseteq \{El(x) \mid x \in S_i\}$ with $|Y_i| \leq k$ such that every element of $\{El(x) \mid x \in S_i\}$ belongs to at least one member of $Y_i$?*

It is easy to observe that MINIMAL-SIZE DFTT′ Problem is a notational variant of MINIMUM COVER Problem, i.e., we take $F = I_C$, $P = \{El(x) \mid x \in S_i\}$ (and therefore $|P| = |S_i|$), and $X = I_C - \{i\}$. Therefore MINIMAL-SIZE DFTT′ Problem is NP-complete. Since the MINIMAL-SIZE DFTT′ Problem is equivalent to the MINIMAL-SIZE DFTT Problem, we conclude that the MINIMAL-SIZE DFTT Problem is also NP-complete. □

According the our previous considerations, the MINIMAL-SIZE DFTT Problem may have to be solved by an optimal ('good') teacher, who is expected to give only relevant information to guarantee fast learning. In this sense our result shows that the task of providing the most useful information for finite identification is NP-complete.

## 6.4 Preset Learning and Fastest Learning

Let us now return to the concept of the *preset learner*. This concept is based on the intuition that it is easier to identify a complicated, large finite structure or an infinite language solely on the basis of their DFTTs, treating those as finite symptoms of the underlying structure. In particular, the use of minimal DFTTs and their influence on the speed of finite identification gives rise to an interesting set of questions. A very natural one is how DFTTs can be used by such preset learning functions. In this section we introduce the notion of *fastest learner* that finitely identifies a language $S_i$ as soon as objective 'ambiguity' between languages has been lifted. In other words, we will define the extreme case of a finite learner who decides on the right language as soon as *any* DFTT has been enumerated.



Let us again take a finitely identifiable class $\mathcal{C}$, and $S_i \in \mathcal{C}$. Now, consider the collection $\mathbb{D}_i$ of all DFTTs of $S_i \in \mathcal{C}$.

**Definition 6.4.1.** *Let $\mathcal{C}$ be an indexed family of recursive sets. $\mathcal{C}$ is* finitely identifiable in the fastest way *if and only if there is a learning function $L$ s.t.:*

$$L(\varepsilon\restriction n) = i \quad \textit{iff} \quad \exists D_i^j \in \mathbb{D}_i\ D_i^j \subseteq \mathrm{set}(\varepsilon\restriction n)\ \&\ \neg\exists D_i^k \in \mathbb{D}_i\ D_i^k \subseteq \mathrm{set}(\varepsilon\restriction n-1).$$

*We will call such $L$ a* fastest learning function.

Intuitively, the fastest learner has to explicitly store all DFTTs of all languages in the given class. Then he makes his conjectures on the basis of the occurrence of the DFTTs in the given text. However, we do not have to provide a set of all DFTTs of all languages explicitly. Rather, we will define them to be accessible via a decision procedure.

**Definition 6.4.2.** *Let $\mathcal{C}$ be an indexed family of recursive sets. The* complete dftt-function *for $\mathcal{C}$ is a recursive function $f_{c\text{-}dftt} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$, such that:*

1. *$f_{c\text{-}dftt}(S,i) = 1$ if and only if $S$ is a DFTT of $S_i$;*
2. *for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, such that $f_{c\text{-}dftt}(S,i) = 1$.*

**Theorem 6.4.3.** *A class $\mathcal{C}$ is finitely identifiable in the fastest way if and only if there is a complete dftt-function for $\mathcal{C}$.*

*Proof.* ($\Rightarrow$) Let us take a class $\mathcal{C}$ and assume that it is finitely identifiable in the fastest way, i.e., there is a learning function $L$ that finitely identifies $\mathcal{C}$, and satisfies the condition of Definition 6.4.1. We define $f : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ in the following way:

$$f(S,i) = \begin{cases} 1 & \text{if } \exists T \subseteq S\ L(\hat{T}) = i, \\ 0 & \text{otherwise.} \end{cases}$$

First, let us observe that $f$ is recursive because $L$ is recursive and there are only finitely many $T \subseteq S$.

Now we have to show that $f$ is a complete dftt-function for $\mathcal{C}$. Let us observe that for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, such that $f(S,i) = 1$. This is so because $L$ finitely identifies $\mathcal{C}$, and so, it finitely identifies an $S_i \in C$ on a text that enumerates $S_i$ in increasing order. It remains to show that:

$$f(S,i) = 1 \text{ iff } S \text{ is a DFTT for } S_i.$$

($\Rightarrow$) Assume that $f(S,i) = 1$, then $\exists T \subseteq S\ L(\hat{T}) = i$. By the definition of fastest learner $L$ we have that $\exists D_i^j \in \mathbb{D}_i\ D_i^j \subseteq \mathrm{set}(\hat{T})$, i.e., there is a DFTT of $S_i$ included in $T$ and hence also in $S$. Since $S \subseteq S_i$, $S$ then has to be a DFTT for $S_i$ as well.

($\Leftarrow$) Assume that $S$ is a DFTT for $S_i$ and, for contradiction, that $f(S,i)=0$. Then it means that $\forall T \subseteq S\ L(\hat{T}) \neq i$. Take $\varepsilon$ any text for $S_i$ and let $\varepsilon' := \hat{S} * \varepsilon$. $\varepsilon'$ is clearly a text for $S_i$, but $L$ is not the fastest learner on $\varepsilon'$. Contradiction.

($\Leftarrow$) Assume that there is a complete dftt-function, $f_{\text{c-dftt}}$, for $\mathcal{C}$. We define $L$ to be the preset learning function based on $f_{\text{c-dftt}}$. Then, by Proposition 6.2.5 $L$ finitely identifies $\mathcal{C}$. We have to show that $L$ is the fastest learner, i.e., for any $S_i \in \mathcal{C}$ and any text $\varepsilon$ for $S_i$:

$$L(\varepsilon\restriction n)=i \quad \text{iff} \quad \exists D_i^j \in \mathbb{D}_i\ D_i^j \subseteq \text{set}(\varepsilon\restriction n)\ \&\ \neg\exists D_i^k \in \mathbb{D}_i\ D_i^k \subseteq \text{set}(\varepsilon\restriction n-1).$$

($\Rightarrow$) Assume that $L(\varepsilon\restriction n)=i$ and, for contradiction, that the right-hand side of the above equivalence does not hold. Then there are two possibilities.

1. $\forall D_i^j \in \mathbb{D}_i\ D_i^j \not\subseteq \text{set}(\varepsilon\restriction n)$. But then from the assumption that $L(\varepsilon\restriction n)=i$, by the definition of $L$ we have also that $\text{set}(\varepsilon\restriction n) \subseteq S_i$ and $\exists S \subseteq \text{set}(\varepsilon\restriction n)$ such that $f_{\text{c-dftt}}(S,i)=1$. Hence, by the definition of $f_{\text{c-dftt}}$, $S$ is a DFTT of $S_i$ and $S \subseteq \text{set}(\varepsilon\restriction n)$. Contradiction.

2. $\exists D_i^k \in \mathbb{D}_i\ D_i^k \subseteq \text{set}(\varepsilon\restriction n-1)$. Since $L(\varepsilon\restriction n)=i$, by the definition of $L$ we have that $\forall \ell < n\ L(\varepsilon\restriction \ell)=\uparrow$ and hence (a) $\forall \ell < n-1\ L(\varepsilon\restriction \ell)=\uparrow$. Moreover, by the definition of $f_{\text{c-dftt}}$, (b) $f_{\text{c-dftt}}(\text{set}(\varepsilon\restriction n-1),i)=1$ and by the definition of DFTT, (c) $\text{set}(\varepsilon\restriction n-1) \subseteq S_i$. From (a), (b) and (c) we can conclude that $L(\varepsilon\restriction n-1)=i$. This contradicts the fact that $L$ is (at most) once defined.

($\Leftarrow$) Assume that for $\varepsilon$ a text for $S_i \in C$ and $n \in \mathbb{N}$ the following holds:

$$\exists D_i^j \in \mathbb{D}_i\ \ D_i^j \subseteq \text{set}(\varepsilon\restriction n)\ \& \tag{1}$$
$$\neg\exists D_i^k \in \mathbb{D}_i\ \ D_i^k \subseteq \text{set}(\varepsilon\restriction n-1) \tag{2}$$

Then, by (2), for all $k<n$ there is no $S$ such that $S \subseteq \text{set}(\varepsilon\restriction k)$ and $f(S,i)=1$. Hence, by the definition of $L$ for all $k<n$, $L(\varepsilon\restriction k)\neq i$. Since $\varepsilon$ is a text for $S_i$ it can not enumerate any DFTT of some different set in $\mathcal{C}$, hence we have that for all $k<n$, $L(\varepsilon\restriction k)=\uparrow$. By (1) and the definition of $f_{\text{c-dftt}}$ we get that $\exists S \subseteq \text{set}(\varepsilon\restriction n)\ f_{\text{c-dftt}}(S,i)=1$. So, $L(\varepsilon\restriction n)=i$.
This completes the proof. □

The above theorem gives a condition of fastest finite identifiability. For any finite set and $i \in \mathbb{N}$, the function $f$ decides whether $S$ is a DFTT of $S_i$. In other words, the class of all DFTTs of $\mathcal{C}$, $\{\mathbb{D}_i \mid S_i \in \mathcal{C}\}$, is uniformly decidable. In fact, the function that the fastest preset learner uses does not have to give a positive answer every time it sees a DFTT, it is enough if the function signals the occurrence of every minimal DFTT.



**Definition 6.4.4.** *Let $\mathcal{C}$ be an indexed family of recursive sets. The* min dftt-function *for $\mathcal{C}$ is a recursive function $f_{\text{min-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$, such that:*

1. *$f_{\text{min-dftt}}(S,i)=1$ if and only if $S$ is a minimal DFTT of $S_i$;*
2. *for every $i \in \mathbb{N}$ there is a finite $S \subseteq \mathbb{N}$, such that $f_{\text{min-dftt}}(S,i)=1$.*

**Theorem 6.4.5.** *A class $\mathcal{C}$ is finitely identifiable in the fastest way iff there is a min-dftt-function for $\mathcal{C}$.*

*Proof.* ($\Rightarrow$) Assume that a class $\mathcal{C}$ is finitely identifiable in the fastest way. Then, by Theorem 6.4.3, there is an effective function $f_{\text{c-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ such that $f_{\text{c-dftt}}(S,i)=1$ iff $S$ is a DFTT for $S_i$ and for every $i \in \mathbb{N}$ there is a finite set $S \subseteq \mathbb{N}$ such that $f_{\text{c-dftt}}(S,i)=1$.

We define $f_{\text{min-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ in the following way:

$$f_{\text{min-dftt}}(S,i) = \begin{cases} 1 & \text{if } f_{\text{c-dftt}}(S,i)=1\ \&\ \neg\exists T \subset S\ f_{\text{c-dftt}}(T,i)=1, \\ 0 & \text{otherwise.} \end{cases}$$

The function $f_{\text{min-dftt}}$ is recursive, because $S$ is finite and $f_{\text{c-dftt}}$ is recursive.
($\Rightarrow$) Assume that there is an effective function $f_{\text{min-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ such that $f_{\text{min-dftt}}(S,i)=1$ iff $S$ is a minimal DFTT for $S_i$ and for every $i \in \mathbb{N}$ there is a finite set $S \subseteq \mathbb{N}$ such that $f_{\text{min-dftt}}(S,i)=1$.

We define $f_{\text{c-dftt}} : \mathcal{P}^{<\omega}(\mathbb{N}) \times \mathbb{N} \to \{0,1\}$ in the following way:

$$f_{\text{c-dftt}}(S,i) = \begin{cases} 1 & \text{if } \exists T \subseteq S\ f_{\text{min-dftt}}(T,i)=1, \\ 0 & \text{otherwise.} \end{cases}$$

The function $f_{\text{c-dftt}}$ is recursive, because $S$ is finite and $f_{\text{min-dftt}}$ is recursive. □

## 6.4.1 Strict Preset Learning

The reader may have expected a stronger notion of preset learner for which a recursive function $F$ exists such that that for each $S_i$, $F(i)$ is a finite set of DFTTs for $S_i$. As announced before we will now be interested in learners that have at their *direct* disposal all minimal DFTTs of languages from the given class. Obviously, such a situation is only generally possible in the case of classes of finite languages. We will consider both possibilities: the one of finite classes of finite languages and that of infinite classes of finite languages. We define strict preset finite identifiability in the following way.

**Definition 6.4.6.** *A finitely identifiable class $\mathcal{C}$ is* strict preset finitely identifiable *iff there is a recursive function $F : \mathbb{N} \to \mathcal{P}^{<\omega}(\mathbb{N})$ such that $F(i)$ outputs the set of all minimal DFTTs of $S_i$.*

**Computational Complexity of Strict Preset Learning** Let us again go back to the case of a finite class of finite sets. To compute the set min-$\mathbb{D}_i$ of all minimal DFTTs of $S_i \in \mathcal{C}$ we need to perform the procedure for finding a minimal DFTT for all possible orderings of elements in $S_i$. Therefore the simple procedure described earlier (in Section 6.3.2) has to be performed $n!$ times. This indicates that finding the set of all minimal DFTTs is in general quite costly. We show that in fact the problem is NP-hard.

**Proposition 6.4.7.** *Finding min-$\mathbb{D}_i$ of $S_i \in \mathcal{C}$ is NP-hard.*

*Proof.* It is easy to observe that once we enumerate $D^i$ of $S_i$, we can find a minimal-size DFTT of $S_i$ in polynomial time by simply picking one of the smallest sets in $D^i$. This means that the MINIMAL-SIZE DFTT Problem for $S_i$ can be polynomially reduced to the problem of finding $D^i$ for $S_i$. □

We will now move to the case of infinite classes of finite sets. We will compare the two notions: strict preset learning and fastest learning in the more general setting of recursive sets. We will proceed with a number of examples. First we will show that there are classes that are finitely identifiable both in the fastest and in the preset way, but that are not strict preset finitely identifiable.

In the following we will use the manner of speech where we will say that $e$ is a Turing machine if we mean that $e$ is an integer that codes a Turing machine and $f(a) = \{b, c\}$ will mean that $f(a)$ codes the finite set containing just $b$ and $c$. Let us recall the notion of Kleene's $T$-predicate.

**Definition 6.4.8** ($T$-predicate (Kleene, 1943)). *$T(e, x, y)$ holds iff $e$ is a Turing machine that on input $x$ performs computation $y$.*

Let us recall that with the use of the $T$-predicate the Halting Problem can be defined in the following way:

$$\exists y T(e, e, y) \iff \varphi_e(e)\downarrow .$$

In other words, the question of whether Turing machine $e$ stops on the input $e$ is equivalent to the question of existence of a computation $y$ performed by $e$ on input $e$.

Let us start with an example of a finitely identifiable class for which a recursive preset learner exists, but which is not strict preset finitely identifiable.

**Theorem 6.4.9.** *There exists a class $\mathcal{C}$ that is finitely identifiable, but for which no recursive function $F$ exists such that for each $i$, $F(i)$ is the set of all minimal DFTTs for $S_i$.*

*Proof.* Let us consider the following class of finite sets $\mathcal{C} = \{S_i \mid i \in \mathbb{N}\}$:

$$S_i = \{2i, 2(\mu y T(i, i, y)) + 1\}.$$



Obviously, the class $\mathcal{C}$ is finitely identifiable. Moreover there exists a recursive fastest learner $L$ that finitely identifies $\mathcal{C}$ and can be defined in the following way:

$$L(\varepsilon\lceil n) = \begin{cases} i & \text{if } 2i \in \text{set}(\varepsilon\lceil n), \\ \mu\ell\ T(\ell, \ell, k) & \text{if } 2k+1 \in \text{set}(\varepsilon\lceil n). \end{cases}$$

We can easily observe that $|S_i| = 2 \iff \varphi_i(i)\downarrow$. Minimal DFTTs of $S_i$ are $\{2i\}$, and, in case $\varphi_i(i)\downarrow$, also $\{2(\mu y T(i, i, y)) + 1\}$. It is clear that no total recursive function $F$ such that

$$g(i) = \{\{2i\}, \{2(\mu y T(i, i, y)) + 1\}\}$$

can be given, because its existence would solve the Halting Problem. □

Therefore, the strict preset learner for the above class cannot be recursive. Since the recursive fastest learner exists (defined in the proof above), the fastest learner cannot be strict preset. Hence, we can conclude that even in the case of classes of finite languages strict preset finite identification is properly included in finite identification and in fastest finite identification. A similar result can be shown for the minimal-size strict preset finite identifiability, i.e., when the learning function requires the set of all the minimal-size DFTTs.

**Proposition 6.4.10.** *There exists a class $\mathcal{C}$ that is finitely identifiable, but for which no recursive function $F$ exists such that for each $i$, $F(i)$ is the set of all minimal-size DFTTs for $S_i$.*

*Proof.* The argument is analogous to the one given in the proof of Theorem 6.4.9. Let $j : \mathbb{N}^2 \to \mathbb{N}$ be a recursive pairing function (bijection) with inverses $j_1$ and $j_2$. We now consider the class $\mathcal{C} = \{S_i \mid i \in \mathcal{C}\}$:

$$S_i = \{3j_1(i), 3j_2(i) + 1, 3(\mu y T(i, i, y)) + 2\}.$$

The set of all minimal-size DFTTs of $S_i$ is $\{\{3(\mu y T(i, i, y)) + 2\}\}$ in case $\varphi_i(i)\downarrow$ and $\{\{3j_1(i)\}, \{3j_2(i) + 1\}\}$ in case $\varphi_i(i)\uparrow$. Therefore the minimal-size DFTTs of $S_i$ cannot be given by a total recursive function, because its existence would solve the Halting Problem. □

## 6.4.2 Finite Learning and Fastest Learning

We have established that recursive fastest identifiability does not require strict preset learnability even in the finite cases. We will now turn to the more general question whether every finitely identifiable class has a fastest learner. The answer is negative—there are finitely identifiable classes of languages which cannot be finitely identified in the fastest way.

**Definition 6.4.11** (Smullyan 1958). *Let $A, B \subset \mathbb{N}$. A separating set is $C \subset \mathbb{N}$ such that $A \subset C$ and $B \cap C = \emptyset$. In particular, if $A$ and $B$ are disjoint then $A$ itself is a separating set for the pair, as is $B$. If a pair of disjoint sets $A$ and $B$ has no computable separating set, then the two sets are* recursively inseparable.

The following theorem says that there are effectively finitely identifiable classes of languages for which there is no effective fastest learner and no recursive function that could enumerate a minimal DFTT for each language.

**Theorem 6.4.12.** *There exists a class $\mathcal{C}$ that is finitely identifiable but is not finitely identifiable in the fastest way (and moreover there is no recursive function that gives a minimal-size DFTT for each language).*

*Proof.* Let $A$ and $B$ be two disjoint r.e. recursively inseparable sets. Let $x \in A$ be equivalent to $\exists y\, Rxy$ with $R$ recursive and let $x \in B$ be equivalent to $\exists y\, Sxy$ with $S$ recursive. We assume that for each $x$ there is at most one $y$ such that $Rxy$ and at most one $y$ such that $Sxy$.

The class of languages $\{S_i \mid i \in \mathbb{N}\}$ is defined as follows:

$$S_i = \{2i, 2i+1\} \cup \{2j \mid Rji\} \cup \{2j+1 \mid Sji\}.$$

The idea is that $\{2i, 2i+1\}$ is the exclusive domain of $S_i$ except that, for some usually much larger $m$, $Rim$ or $Sim$ may be true, and then $2i \in S_m$ or $2i+1 \in S_m$, respectively. There can be at most one such $m$, and for that $m$ only one of $Rim$ or $Sim$ can be true. However, since $A$ and $B$ are recursively inseparable there can be no recursive function $f$ that makes this latter choice for each $i$.

It is clear that to decide definitely that the language is $S_i$ it suffices to encounter $2i$ and $2i+1$, so $\{2i, 2i+1\}$ is a DFTT for $S_i$. But, if $i \in A$, then $\{2i+1\}$ is a DFTT for $S_i$, and if $i \in B$, then $\{2i\}$ is a DFTT for $S_i$. To be more precise, $\{2i+1\}$ is a DFTT for $S_i$ if $i \notin B$ and $\{2i\}$ is a DFTT for $S_i$ if $i \notin A$. But no recursive learner can decide on this, so there cannot be a recursive fastest learner, or a function that gives, for each $i$, a minimal DFTT for $S_i$. □

In general there is no reason for preset learners to use only minimal DFTTs. They can have a set of DFTTs for each $S_i$ and only use those. The class given in the proof of Theorem 6.4.12, for which no recursive fastest learner is possible can obviously have a preset learner but the DFTTs do not include the minimal ones.

The above theorem shows that fastest finite identifiability is properly included in finite identification and hence also in preset finite identification. Hence, we have shown the existence of yet another kind of learning, even more demanding than finite identification. Speaking in terms of conclusive update, our considerations show that in some cases, even if computable convergence to certainty is possible, it is not computable to conclude that at the first moment in which objective ambiguity disappears.



In the light of these discoveries about preset learning, we can give a computational justification for introducing multi-agency to this setting. It seems to be justifiable to switch the perspective from the single agent, learning-oriented view, to the two agent game of learner and *teacher*. The responsibility of effective learning, in the line with natural intuitions, is in the hands of the teacher, whose computational task is to find samples of information that guarantee optimal learning. Intuitively, it is not very surprising that the task of finding such minimal samples can be more difficult than the complexity of the actual learning. As such, computing the minimal(-size) DFTTs seems to go beyond the abilities of the learner and is not necessary in order to be rational or successful. However, such a task is natural to be performed by a teacher.

## 6.5 Conclusions and Perspectives

We used the characterization of finite identification of languages from positive data to discuss the complexity of optimal learning and teaching strategies in finite identification. We introduced two notions: minimal DFTT and minimal-size DFTT. By viewing the informativeness of examples as their power to eliminate certain conjectures, we have checked the computational complexity of 'finite teachability' from minimal DFTT and minimal-size DFTT. In the former case the problem turns out to be PTIME computable, while the latter falls into the NP-complete class of problems. This suggests that it is easy to teach in a way that avoids irrelevant information but it is potentially difficult to teach in the most effective way. We also conceptually extended the characterization of finite identification and introduced the notion of preset learner. We compared variations of the latter with the idea of fastest finite identification. In particular, we focused on the notion of strict preset finite identifiers that have at their disposal all minimal DFTTs of every language in the class. Even in the setting of classes of finite sets this type of learning turns out to be restrictive with respect to finite identifiability. We have exhibited a recursively finitely identifiable class that cannot be recursively finitely identifiable in the fastest way. Hence we have established a new more restrictive kind of finite identification.

Links between finite identification and dynamic epistemic logic have already been established and described in Chapters 3–5. For dynamic epistemic logic the restriction to finite sets of finite languages is very natural, so our analysis of its complexity can be applied to strengthen this connection. The complexity of fastest finite identification corresponds to the complexity of fastest conclusive update in dynamic epistemic logic. It gives a new measure of computational complexity of certainty— a measure that corresponds to the question of how difficult it is to reach the state of irrevocable knowledge.

The main assumptions of learning theory and hence also of the present chapter is the cooperative nature of the interaction between the learner and the teacher. In the next chapter we will reflect upon other possible learning attitudes.