Chapter 7

Many-one Reduction

Reduction is one of the most central concepts of the theory of computability. Informally, problem X reduces problem Y just in case there exists a way to transform an arbitrary solution to X into a solution to Y.

If X reduces problem Y, then up to computability, Y is no harder than X, for there is no way that Y could be unsolvable if X is solvable. Thus, one writes

$$X \geq Y$$

iff X reduces Y. So reducibility provides a notion of *relative difficulty* among problems.

You may then wonder if, for example, there is a "hardest" problem among, say, the r.e. sets. Such problems are said to be **r.e.-complete**. A decision procedure for an r.e.-complete problem would amount to a solution to every r.e. problem.

Reduction is important because it allows you to turn solutions to one problem into solutions to another. If you know that $X \geq Y$ and you have a decision procedure for X, the reduction yields a decision procedure for Y.

More importantly, if $X \geq Y$ and you can argue that Y is not decidable, then neither is X. So reduction is a handy way to prove that a problem is unsolvable: just show that it reduces a problem already known to be unsolvable. So far, your only example of an unsolvable problem is the halting problem. The proof was by diagonalization. That worked because the halting problem is concocted to be the counter-diagonal of the characteristic matrix of the r.e. sets. Now you can use reduction to prove that more intuitively presented problems are not effectively decidable.

7.1 Definitions

Many-one reducibility is defined as follows:

$$X \leq_m Y \iff (\exists \text{ total recursive } g)(\forall x)\ (X(x) \iff Y(g(x))).$$

Then say that Y **many-one reduces** X, or that X is **many-one reducible** to Y. The definition says that the total function g projects X into Y and $\neg X$ into $\neg Y$. Use of the "less than" notation invites the challenge that such notation is justified.

Proposition 7.1

$\leq_m$ is a pre-order (reflexive and transitive).

$X \leq_m Y \iff \overline{Y} \leq_m \overline{X}$.

Exercise 7.1 *Prove it.*

Symmetrizing a pre-order always yields a natural concept of equivalence:

$$X \equiv_m Y \iff X \leq_m Y \wedge Y \leq_m X.$$

This is called **many-one equivalence**. Equivalence relations allow us to define equivalence classes called **many-one degrees**.

$$[X]_m = \{Y \subseteq \mathbf{N} | X \equiv_m Y\}.$$

Another mathematical instinct is to lift the original pre-order to define a partial order on equivalence classes:

$$[X]_m \leq_m [Y]_m \iff X \leq_m Y.$$

But this kind of talk can result in contradictions pretty fast unless you show that the relation $[X]_m \leq_m [Y]_m$ doesn't depend on the representatives X, Y used to pick out the respective classes. In other words, you must show that $\leq_m$ is indeed a partial order over degrees. If you haven't already done this in your discrete math class (or if you didn't care about it then), do it right now.

Now let $\Gamma \subseteq Pow(\mathbf{N})$. Then Define:

$$X \text{ is } \Gamma\textbf{-hard} \iff (\forall Y \in \Gamma)\ Y \leq_m X.$$

$$X \text{ is } \Gamma\textbf{-complete} \iff X \in \Gamma \wedge X \text{ is } \Gamma\textbf{-hard}.$$

7.2 R.e. Completeness

Recall that a set X is m-complete in a collection of problems just in case X is in the collection and each member of the class m-reduces to X. It turns out that K is m-complete among the r.e. sets. That sounds like a hard thing to prove: how can you construct infinitely many m-reductions all at once? In this case, it's surprisingly easy.

Proposition 7.2 *K is r.e. complete.*

Proof. You need to reduce each r.e. set to K. Suppose S is r.e., so for for some k, $S = W_k$. Define

$$\begin{aligned} \psi(x,y) &\simeq (\mu z)\ U(k,(z)_0,(z)_1,\langle x\rangle) \\ &\simeq \phi_k(x). \end{aligned}$$

Choose an index and apply s-m-n to obtain total recursive g such that:

$$\phi_{g(x)}(y) \simeq \psi(x,y).$$

Thus,

$$\begin{aligned} S(x) &\iff W_k(x) \\ &\iff \phi_k(x)\downarrow \\ &\iff \psi(x,g(x))\downarrow \\ &\iff \phi_{g(x)}(g(x))\downarrow \\ &\iff K(g(x)). \end{aligned}$$

$\dashv$

7.3 Preservation Results

The main applications of reducibility theory are consequences of the elementary but important fact that recursive enumerability and recursiveness are preserved downward in the many-one ordering. The idea is that the reduction is a "pre-processor" that "converts" the new problem X into inputs for the problem Y in such a way that the solution to Y is converted into a solution to X.

Proposition 7.3

1. $X \leq_m Y \wedge Y$ is r.e. $\Rightarrow X$ is r.e.
2. $X \leq_m Y \wedge Y$ is recursive $\Rightarrow X$ is recursive.

Proof. Suppose $X \leq_m Y$. Then

$$(\exists \text{ total recursive } g)(\forall x)\ X(x) \iff Y(g(x)).$$

Suppose Y is recursive. Then $Y(x)$ is a recursive function of x. So $X(x) = Y(g(x))$ is a recursive function of x. So X is recursive.
Suppose Y is r.e. So for some n, $Y = W_n = dom(\phi_n)$. The composition $C(\phi_n, g)$ is also partial recursive. You have

$$\begin{aligned} C(\phi_n,g)(x)\downarrow &\iff \phi_n(g(x))\downarrow \\ &\iff Y(g(x)) \\ &\iff X(x). \end{aligned}$$

Thus, $X = dom(C(\phi_n, g))$, so X is r.e.$\dashv$

7.4 Examples

Now you may use the preceding result to show that lots of new problems are unsolvable, by reducing K or $\neg K$. For example, consider the following, elementary input-output properties of partial recursive indices.

$$\begin{aligned} Zd(x) &\iff \phi_x = o; \\ Tot(x) &\iff W_x = \mathbf{N}; \\ Und(x) &\iff W_x = \emptyset; \\ Inf(x) &\iff W_x \text{ is infinite}; \\ Onto(x) &\iff E_x = \mathbf{N}. \end{aligned}$$

In general, let $\Gamma \subseteq Part$ be given. Define

$$index(\Gamma) = \{n \in \mathbf{N} : \phi_n \in \Gamma\}.$$

Now say that X is an **index set** just in case there exists a $\Gamma \subseteq Part$ such that $X = index(\Gamma)$. All the above are index sets.

Exercise 7.2 *Is K an index set?*

Proposition 7.4

$K \leq_m Zd, \neg Und, Tot, \neg Tot, Inf, \neg Inf$;

$\neg K \leq_m \neg Zd, Und, Tot, \neg Tot, Inf$.

Proof. The second statement follows from the first by proposition 7.1.

For the first statement, the idea is to produce a very informative reduction. For example, suppose you can construct a total recursive g such that that

$$\phi_{g(i)} = \begin{cases} o & \text{if } i \in K; \\ \emptyset & \text{if } i \notin K. \end{cases}$$

Such a g reduces K to Zd, $\neg Und$, Tot and Inf all at once! For example:

$$\phi_{g(i)} \in \begin{cases} Zd & \text{if } i \in K; \\ \neg Zd & \text{if } i \notin K, \end{cases}$$

and similarly for $\neg Und$, Tot, and Inf.

So how do you produce such a function? Since you are trying to construct a total recursive function of indices, you will expect that you will use the universal construction together with the *s-m-n* theorem. Start with the universal construction:

$$\begin{aligned} \psi(i,x) &\simeq o(\mu z.U(i,(z)_0,(z)_1,\langle i \rangle)) \\ &\simeq \begin{cases} 0 & \text{if } \phi_i(i)\downarrow; \\ \uparrow & \text{otherwise} \end{cases} \\ &\simeq \begin{cases} 0 & \text{if } i \in K; \\ \uparrow & \text{otherwise.} \end{cases} \end{aligned}$$

Then apply the usual *s-m-n* sequence to obtain total recursive g such that

$$\begin{aligned}\phi_{g(i)}(x) &\simeq \psi(i,x)\\ &\simeq \begin{cases} 0 & \text{if } i \in K;\\ \uparrow & \text{otherwise.}\end{cases}\end{aligned}$$

This total recursive g witnesses,

$$K \leq_m Zd, \neg Und, Tot, Inf.$$

To reduce K to $\neg Zd$, $\neg Tot$, and $\neg Inf$, you need a sneakier strategy. The preceding construction went "with the grain", halting just in case a given index halts on itself. It's easy to make $\phi_{g(i)}(x)$ halt when $\phi_i(i)$ halts. This time, we need $\phi_{g(i)}$ to do a lot of halting when $\phi_i(i)$ does *not* halt. That sounds suspicious. Doesn't halting when something else doesn't halt presuppose a solution to the halting problem? Not quite. The trick is that $\phi_{g(i)}(x)$ can halt with 0 on input x when $\phi_i(i)$ does not halt *under resource bound* x. Using the universal construction, define the partial recursive function:

$$\begin{aligned}\psi(i,x) &\simeq (\mu z)\ (\forall y \leq x)\ \neg U(i,x,y,\langle i\rangle)\\ &\simeq \begin{cases} 0 & \text{if } (\forall y \leq x)\ \neg U(i,x,y,\langle i\rangle);\\ \uparrow & \text{otherwise.}\end{cases}\end{aligned}$$

Now apply the usual *s-m-n* sequence to arrive at a total recursive g such that:

$$\begin{aligned}\phi_{g(i)}(x) &\simeq \psi(i,x);\\ &\simeq \begin{cases} 0 & \text{if } (\forall y \leq x)\ \neg(U(i,x,y,\langle i\rangle));\\ \uparrow & \text{otherwise.}\end{cases}\end{aligned}$$

Notice that if $\phi_i(i) \uparrow$ then the first condition is met for all x and if $\phi_i(i) \downarrow$ then there is some y such that the first condition is not met for any $x \geq y$. Hence,

$$\begin{aligned}\phi_{g(i)} &\simeq \begin{cases} o & \text{if } \phi_i(i) \uparrow;\\ \text{some finite domain function} & \text{otherwise.}\end{cases}\\ &\simeq \begin{cases} o & \text{if } i \notin K;\\ \text{some finite domain function} & \text{if } i \in K.\end{cases}\end{aligned}$$

Hence, g witnesses that K is reducible to $\neg Zd$, $\neg Tot$, $\neg Inf$. $\dashv$

Corollary 7.5

$\neg Zd, Und, Tot, \neg Tot, Inf, \neg Inf$ are not r.e.

$Zd, \neg Zd, Und, \neg Und, Tot, \neg Tot, Inf, \neg Inf$ are not recursive.

Proof. Propositions 7.3, 7.4$\dashv$

Exercise 7.3 *Show that $Onto, \neg Onto$ are not r.e. Hint: use slight modifications of the preceding constructions to reduce both K and $\neg K$ to $Onto$. Instead of returning 0 when the relevant condition is condition is met, return x.*

7.5 Recursive Inseparability

Suppose that X, Y are subsets of $\mathbf{N}$. We say that X and Y are **recursively inseparable** just in case there exists no recursive set S such that $X \subseteq S$ and $S \cap Y = \emptyset$. In other words, there is no way to draw a "recursive line" between X and Y. So of course if X and Y are not disjoint, then they are recursively inseparable. Recursive inseparability is interesting, because each recursive inseparability result yields a whole family of non-recursiveness results all at once. The smaller X and Y are, the informative it is to know that X and Y are recursively inseparable.

Exercise 7.4 *Show that if X and Y are recursively inseparable and X', Y' are disjoint and $X \subseteq X'$ and $Y \subseteq Y'$ then X', Y' are recursively inseparable.*

Exercise 7.5 *Show that if there is a total recursive f that sends every member of K into X and every member of $\neg K$ into Y then X, Y are recursively inseparable.*

Exercise 7.6 *Show that $\{n : \phi_n = p_1^1\}$ and $\{n : \phi_n = \emptyset\}$ are recursively inseparable. Hint: use the preceding exercises.*

7.6 Rice's Theorem

(Rogers exercise 5-29) In the preceding result, none of the index sets in question turned out to be recursive. In fact, *no nontrivial index set is recursive.* In practical terms, that means that *a compiler of a general programming language cannot reliably check for any input-output property of the program being compiled.* The following proposition establishes these facts. To facilitate the statement of the theorem, say that a set is **nontrivial** just in case it is neither $\mathbf{N}$ nor $\emptyset$. The idea is that to determine any nontrivial set theoretic property of ϕ_i, one would have to determine, for some x, whether $\phi_i(x) \downarrow$, but one would need special powers to infallibly conclude that $\phi_i(x) \uparrow$.

Proposition 7.6 Rice's theorem *No nontrivial index set is recursive.*

Proof. Let Γ be a nontrivial subset of $Part$.
Case I: suppose $\emptyset \in \Gamma$.
Then since Γ is nontrivial, there exists some $\delta \in Part - \Gamma$. Now construct:

$$\psi(x, y) \simeq \delta(y) + o((\mu z)\ U(x, (z)_0, (z)_1, \langle x \rangle)).$$

By using *s-m-n* in the usual way, construct total recursive g such that

$$\phi_{g(x)} \simeq \delta(y) + o((\mu z)\ U(x, (z)_0, (z)_1, \langle x \rangle)).$$

Thus,

$$\begin{array}{rcccl} K(x) & \Rightarrow & \phi_{g(x)} = \delta \notin \Gamma & \Rightarrow & g(x) \in index(\Gamma); \\ \neg K(x) & \Rightarrow & \phi_{g(x)} = \emptyset \in \Gamma & \Rightarrow & g(x) \notin index(\Gamma). \end{array}$$

So $\neg K \leq_m index(\Gamma)$. Thus, $index(\Gamma)$ is not r.e. and hence is not recursive.

Case II: suppose $\emptyset \notin \Gamma$. Then since Γ is nontrivial, there exists some $\delta \in Part - \Gamma$. So the same reduction g establishes that $K \leq_m index(G)$, so $\neg index(\Gamma)$ is not r.e. and hence $index(\Gamma)$ is not recursive.$\dashv$

7.7 The Rice-Shapiro Theorem

(Rogers Exercise 5.37)

There are no interesting recursive index sets. But there are clearly interesting r.e. index sets; $\neg Und$, for example (do you see why this is r.e.?) Can we obtain a nice, general characterization of the r.e. index sets? Yes! And the result is very interesting from an epistemological point of view. We need a couple of natural concepts to do so. Let θ be a partial recursive function. Define

$$[\theta] = \{\psi \in Part : \theta \subseteq \psi\}.$$

That is, $[\theta]$ denotes the set of all partial recursive functions that extend $[\theta]$. Say that

$$\Gamma \text{ is } \textbf{experimentally verifiable} \iff$$

$$\text{there exists a set } \Delta \text{ of finite functions such that } \Gamma = \bigcup_{\theta \in \Delta} [\theta].$$

Why do I call this "experimental verifiability"? Well, suppose that Γ is experimentally verifiable in the sense just defined. Suppose you investigate whether $x \in index(\Gamma)$ by treating x as a black box and simulating index x in a dovetail construction to discover more and more values of ϕ_x through time. Then whenever you observe that the data points you have received extend some $\theta \in \Delta$, you know that the function you are experimentally studying is in Γ, because every extension of a finite function in Δ is in Γ. Conversely, if the data never extend a finite function in Δ, then the function you are studying is not in Γ, so you are correct never to declare with certainty that the function you are studying is in Γ.

The following result provides a more logically explicit way of looking at experimental verifiability.

Lemma 7.7

Γ *is experimentally verifiable* $\iff$ $(\forall \psi \in Part)\ \psi \in \Gamma \Leftrightarrow (\exists$ *finite* $\theta \subseteq \psi)\ (\theta \in \Gamma)$.

Proof. Suppose Γ is experimentally verifiable.
Thus, there exists a collection Δ of finite functions such that $\Gamma = \bigcup_{\theta \in \Delta} [\theta]$.
Let $\psi \in Part$ be given.
Suppose that $\psi \in \Gamma$.
Then for some finite $\theta \in \Delta, \psi \in [\theta] \subseteq \Gamma$.
Hence, $\theta \subseteq \psi$.
Since $\theta \in [\theta]$, we have $\theta \in \Gamma$.

Conversely, suppose $(\exists \text{ finite } \theta \subseteq \psi)\theta \in \Gamma$.
So for some finite $\tau \in \Delta, \theta \in [\tau] \subseteq \Gamma$.
So $\tau \subseteq \theta \subseteq \psi$
So $\psi \in [\tau] \subseteq \Gamma$.

For the converse of the lemma, suppose that

$$(\forall \psi \in Part)\ (\psi \in \Gamma \Leftrightarrow (\exists \text{ finite } \theta \subseteq \psi)\ \theta \in \Gamma).$$

For each $\psi \in \Gamma$, choose such a θ_ψ.
Define $\Delta = \{\theta_\psi : \psi \in \Gamma\}$.
Since each $\psi \in \Gamma$ is included in $[\theta_\psi]$, we have: $\Gamma \subseteq \bigcup_{\theta \in \Delta}[\theta]$.
Also, $[\theta_\psi] \subseteq \Gamma$, by choice of $[\theta_\psi]$, so $\bigcup_{\theta \in \Delta}[\theta] \subseteq \Gamma$.
Thus, $\Gamma = \bigcup_{\theta \in \Delta}[\theta]$.$\dashv$

The following result says that if formal methods can verify a property of input-output behavior by peering at the programs, then an ideal agent would have been able to verify the input-output property from experimental evidence derived from studying the program as a black box on various inputs for various amounts of runtime. This is an amazing fact. *A priori*, one would expect to do a lot better by looking at code than by simply doing behavioral experiments. Properties that are not experimentally verifiable are exactly the properties that pose Hume's problem of inductive skepticism. Thus, we see that some index sets are not formally verifiable because of empirical skepticism. This seems to cut to the heart of the traditional empiricist distinction between *relations of ideas and matters of fact*. The idea was that relations of ideas are relatively unproblematic since the mind has complete control of the ideas "all at once", whereas empirical truth always outruns our observations. The following result shows that the same is actually true in formal reasoning by computational means.

Proposition 7.8

Γ *is not experimentally verifiable* $\Rightarrow \neg K \leq_m index(\Gamma)$.

Hence, $index(\Gamma)$ *is r.e.* $\Rightarrow \Gamma$ *is experimentally verifiable*

Proof. Suppose that Γ is not experimentally verifiable. Then by the preceding lemma,

$$\neg(\forall \psi \in Part)\ \psi \in \Gamma \Leftrightarrow (\exists \text{ finite } \theta \subseteq \psi)\ \theta \in \Gamma.$$

Driving the negation in yields $(\exists \psi \in Part)$ such that:

$$\begin{array}{rl} \text{Case I:} & \psi \in \Gamma \wedge (\forall \text{ finite } \theta \subseteq \psi)\ \theta \notin \Gamma. \\ & \textit{or} \\ \text{Case II:} & \psi \notin \Gamma \wedge (\exists \text{ finite} \theta \subseteq \psi)\ \theta \in \Gamma. \end{array}$$

Case I: Consider the situation. Some $\psi \in \Gamma$ has the unfortunate property that each finite evidence sequence drawn experimentally from ψ is the complete evidence for a non-member of Γ. That's just the problem of induction, which tells us that empirical verifiability is impossible. Now we are going to make the

problem of induction into a problem for formal program verification. That is, we are going to cross the philosophical boundary between matters of fact and relations of ideas!

In particular, we are going to reduce $\neg K$ to $index(\Gamma)$. So

- given an index x such that $\neg K(x)$, we want to produce an index that acts like ψ, and
- given an index x such that $K(x)$, we want to produce an index that acts like some finite subfunction of ψ.

Intuitively, we can do this by means of a universal construction that simulates $\phi_x(x)$ for longer and longer runtimes waiting to see it halt. As more and more time passes, the construction produces more and more of the outputs specified by ψ, so that if $\phi_x(x)$ never halts, we produce all of ψ. On the other hand, if $\phi_x(x)$ halts, we use this as a signal to stop making new outputs, so we end up producing some finite subfunction of ψ. Notice that his reduction "goes against the grain", turning halting into non-halting (check the example above to see how to set up such a construction).

Case II: In this situation, there is a function ψ *not* in Γ such that some finite subfunction θ of ψ is in Γ. Again, we wish to reduce $\neg K$ to $index(\Gamma)$. We can do this by a universal construction that pretends to be ??? until $\phi_x(x)$ halts and that starts making outputs from ψ thereafter. . . .$\dashv$

Exercise 7.7 *Finish cases I and II. Hint: follow the intuitive motivation, inspect the reductions involved in proposition 7.4, and give proper constructions using the universal and* s-m-n *theorems.*

Exercise 7.8 *Use lemma 7.7 and proposition 7.8 to show that the set of all indices of functions with values never exceeding k is not r.e. If you feel shaky, try a few of the problems mentioned in corollary 7.5.*

Experimental verifiability is only a necessary condition for recursive enumerability of an index set.

Exercise 7.9 *Find a counterexample to the converse of proposition 7.8. Hint: Make the property "empirically" easy to decide but make it computationally as hard as $\neg K$ to "interpret" the readily available data.*

That is because the "verifying data sequences" in Δ might not be effectively enumerable. To obtain necessary and sufficient conditions for index(Γ) to be r.e., we can effectively encode finite functions as numbers and then require that the set of code numbers be r.e.

Let Fin denote the set of all finite, partial, unary functions on the natural numbers. Each such function can be represented as a finite sequence of values, possibly with some undefined places. It doesn't hurt to have multiple indices for each function, so there is no need to worry about including some undefined places at the end of the sequence.

$$(3, 5, \uparrow, 7, \uparrow).$$

This can be converted into a sequence of natural numbers by encoding $\uparrow$ as 0 and defined value n as $n+1$ and then taking the Gödel number of the resulting sequence:

$$\langle 4, 6, 0, 8, 0 \rangle.$$

Let θ_n denote the finite function whose code number according to this scheme is n. Hence,

$$\begin{aligned} \theta_n(x) &\simeq \begin{cases} (n)_x \dot{-} 1 & \text{if } x < lh(n) \wedge (n)_x > 0; \\ \uparrow & \text{otherwise.} \end{cases} \\ &\simeq (n)_x \dot{-} 1 \cdot (\mu z)\ (x < lh(n) \wedge (n)_x > 0). \end{aligned}$$

Note that the minimalization does not depend on z, so its value is either zero or $\uparrow$.

It is possible to move effectively from finite indices to partial recursive indices, but not conversely. The intuitive idea behind this is that one can decide which pairs are in a finite function from its *ff* index but not from an arbitrary partial recursive index.

Proposition 7.9

($\exists$ *total recursive* f) ($\forall n$) ($\phi_{f(n)} = \theta_n$).

Proof. You need total recursive f such that for each n, x:

$$\begin{aligned} \phi_{f(n)}(x) &\simeq \theta_n(x) \\ &\simeq (n)_x \dot{-} 1 \cdot (\mu z)\ (x < lh(n) \wedge (n)_x > 0) \\ &\simeq \psi(n, x). \end{aligned}$$

Since ψ is partial recursive, obtain f in the usual way by applying the s-m-n theorem to ψ. $\dashv$

Given a set Δ of finite functions, define

$$\textit{ff-index}(\Delta) = \{n | \theta_n \in \Delta\}.$$

Define:

Γ is **effectively experimentally verifiable** just in case there exists a collection Δ of finite functions such that

1. *ff-index*(Δ) is r.e. and
2. $\Gamma = \bigcup_{\theta \in \Delta} [\theta]$

Now we can state an exact characterization of the r.e. index sets in terms of effective experimental verification. So formal program verification is *essentially the same as* effective empirical program verification.

Proposition 7.10 *index*(Γ) *is r.e.* $\iff$ Γ *is effectively experimentally verifiable.*

Proof. Suppose the right-hand-side of the proposition is true. So we may choose k so that $\mathit{ff}\text{-}index(\Delta) = W_k$. To verify $index(\Gamma)$, we use a dovetail construction to check whether the given index extends some finite function in Δ. We can verify this because we can decide which pairs are in a finite function from ff indices. That is the whole point of introducing ff indices.
Define

$$\psi(n) \simeq (\mu z)\ (\forall w \leq lh((z)_0))\ U(n, (z)_1(((z)_0)_w)_0, \langle((((z)_0)_w)_1\rangle.$$

This construction seeks pairs $\langle i, j\rangle$, where i is a runtime bound and j is viewed as an ff-index. Then by the definition of the ff indexing, $(((z)_0)_w)_0$ denotes an element of the domain of θ_j and $(((z)_0)_w)_1$ denotes the corresponding value of θ_j. So on input n, the construction simply searches for an ff index j such that $\theta_j \subseteq \phi_n$. There is such a j just in case

$$\phi_n \in \bigcup\nolimits_{\theta\in\Delta}[\theta] = \Gamma.$$

So we have a partial recursive ψ such that $index(\Gamma) = dom(\psi)$.

Conversely, suppose that $index(\Gamma)$ is r.e. We need to construct a collection of finite functions such that $\mathit{ff}\text{-}index(\Delta)$is r.e. So let $index(\Gamma) = W_n$. Then by proposition 7.8 and lemma 7.7 we have

$$(\forall\psi \in Part)\ (\psi \in \Gamma \Leftrightarrow (\exists \text{ finite } \theta \subseteq \psi)\theta \in \Gamma). \tag{7.1}$$

Let some total recursive g translate ff indices into partial recursive indices as promised by proposition 7.9. Define:

$$\Delta = \{\theta_k | W_n(g(k))\}.$$

Now define

$$\psi(k) \simeq \phi_n(g(k)).$$

Evidently,

$$\mathit{ff}\text{-}index(\Delta) = dom(\psi).$$

so $\mathit{ff}\text{-}index(\Delta)$ is r.e. Since $index(\Gamma) = W_n$, we have $\Delta \subseteq \Gamma$. By $(*)$, we have that each $\psi \in \Gamma$ extends some element of Δ. Thus $\Gamma = \bigcup_{\theta\in\Delta}[\theta]$. So Γ is effectively experimentally verifiable.$\dashv$