Chapter 4

The Grzegorczyk Hierarchy

4.1 Reprise

The primitive recursive functions are defined as the closure of R and C over a basic stock of functions. Both the logician's diagonal argument and the double-recursion diagonal arguments involve functions that can "simulate" n nested primitive recursions when given an argument n. Intuitively, the situation is that each primitive recursive function is allowed only finitely many nested primitive recursions, so functions that require arbitrarily many (without bound) are not primitive recursive.

This situation calls out for greater attention to which functions can be defined with a given, fixed number of primitive recursion operations. In other words, it makes sense to look at the hierarchy of increasing classes E_n defined in terms of increasing numbers of primitive recursions, where each class is closed under composition. Every primitive recursive function will be at some finite level of the hierarchy:

$$Prim = \bigcup_x E_x$$

A natural idea would be to start with the basic functions closed under composition and then to add in successively more complex functions like addition, multiplication, exponentiation, etc., closing under composition each time.

But composition is a little weak. Recall that some slow-growing functions like cutoff subtraction required the R schema for their definitions even though they don't really use it to generate faster growth. We would like to distinguish *growth-generating* applications of R from applications that don't generate growth.

We can get around the difficulty by closing each class under *bounded recursion* as well as composition. An application of R is bounded in a class of functions if the function resulting from the application is bounded everywhere by some other function already established to be in the class.

Bounded Recursion An application of R to g, f is bounded just in case there is a *previously defined* h such that for all $\vec{x}$

$$R(g, f)(\vec{x}) \leq h(\vec{x})$$

For example, in the cases of decrementation and cutoff subtraction, we have

$$\begin{aligned} Dec(x) &< p_1^1(x) \\ x \dot{-} y &< p_2^1(x, y) \end{aligned}$$

So both of these functions are definable by bounded recursion and composition from basic functions. Similarly for quotient, remainder, sg, $\overline{sg}$, min, and max.

4.2 The Grzegorczyk Hierarchy

We first introduce a "spine" $\{e_n | n \in \mathbf{N}\}$ of representative functions requiring increasing numbers of primitive recursions for their definition. It doesn't matter that much which spine we choose, since the closure operations and bounded recursion will smooth out the resulting classes, yielding invariance. We will follow Rose's development (pp. 31-36), keeping in mind that there are others. Define:

$$\begin{aligned} e_0(x, y) &= x + y \\ e_1(x) &= x^2 + 2 \\ e_{n+2}(0) &= 2 \\ e_{n+2}(x + 1) &= e_{n+1}(e_{n+2}(x)) \end{aligned}$$

Then define:

$E_0 =$ the least set X containing the basic functions
and closed under composition and bounded recursion
$E_{n+1} =$ the least set X containing the basic functions, e_n,
and closed under composition and bounded recursion

Then the indexed collection

$$\{E_n | n \in \mathbf{N}\}$$

is called the **Grzegorczyk hierarchy**.
Note that by the third level the higher functions iterate the lower functions on argument 2.

Proposition 4.1 $e_{n+3}(x) = (e_{n+2})^x(2)$.

Proof:

$$\begin{aligned} e_{n+3}(0) &= 2 \\ &= (e_{n+2})^0(2) \end{aligned}$$

$$\begin{aligned} e_{n+3}(x) = (e_{n+2})^x(2) \Rightarrow e_{n+3}(x+1) &= e_{n+2}(e_{n+3}(x)) \\ &= e_{n+2}((e_{n+2})^x(2)) \\ &= (e_{n+2})^{x+1}(2). \end{aligned}$$

⊣

4.3 Double Induction

Many of the basic results about double recursion are proved by double induction. As you know very well, induction is the schema:

$$[\Phi(0) \wedge \forall x(\Phi(x) \rightarrow \Phi(x+1))] \rightarrow \forall x \Phi(x) \tag{4.1}$$

What if we wish to prove that for all x, $\forall y \Psi(x, y)$?
If we substitute the desired conclusion $\forall y \Psi(x, y)$ for $\Phi(x)$, then we obtain:

$$[\forall y \Psi(0, y) \wedge \forall x((\forall y \Psi(x, y)) \rightarrow (\forall y \Psi(x+1, y)))] \rightarrow \forall x(\forall y \Psi(x, y)) \tag{4.2}$$

To get the first conjunct of the antecedent of 3.1, it suffices (by induction) to prove:

$$\Psi(0, 0) \wedge \forall y(\Psi(0, y) \rightarrow \Psi(0, y+1)) \tag{4.3}$$

To get the second conjunct, we can use induction to get the universal quantifier on y in the consequent, so it suffices to show:

$$\forall x((\forall y \Psi(x, y)) \rightarrow \big[\Psi(x+1, 0) \wedge [\Psi(x+1, y) \rightarrow \Psi(x+1, y+1)]\big] \tag{4.4}$$

Double Induction Schema

DI 1. $\Phi(0, 0) \wedge$
DI 2. $\forall y[(\Phi(0, y) \rightarrow \Phi(0, y+1))] \wedge$
DI 3. $\forall x((\forall y \Phi(x, y)) \rightarrow$
  a. $[\Psi(x+1, 0) \wedge$
  b. $[\Psi(x+1, y) \rightarrow \Psi(x+1, y+1)]] \rightarrow \forall x \forall y \Psi(x, y)$

The technique is illustrated by the following result which will be of use shortly.

Proposition 4.2 *for all $n \geq 1$,*

1. $e_n(x) \geq x + 1$
2. $e_n(x + 1) > e_n(x)$
3. $e_{n+1}(x) \geq e_n(x)$
4. $(e_n)^k(x) \leq e_{n+1}(x + k)$.

Proof. 1. By double induction. Note: to fit the problem into the double induction schema, we rewrite the formula as

$$e_{n+1}(x) \geq x + 1$$

DI 1, 2:

$$\begin{aligned} e_{0+1}(x) &= x^2 + 2 \\ &\geq x + 1. \end{aligned}$$

DI 3:
suppose $\forall x(e_{n+1}(x) \geq x + 1)$.
DI 3.a:

$$\begin{aligned} e_{n+2}(0) &= 2 \\ &\geq 0 + 1. \end{aligned}$$

DI 3.b:
suppose $e_{n+2}(x) \geq x + 1$. Then

$$\begin{aligned} e_{n+2}(x + 1) &= e_{n+1}(e_{n+2}(x)) \\ &\geq e_{n+2}(x) + 1 \\ &\geq x + 2. \end{aligned}$$

⊣

Exercise 4.1 *Prove the remaining clauses of the proposition. The strict inequality in 4.2.2 will be used below, so I mean it.*

4.4 The Elementary Functions (Kalmar)

The collection of elementary functions is defined by:

$$E = E_3$$

It is often repeated that almost all number theoretical functions that arise in practice are elementary. This isn't surprising when one considers that composed exponentials ($2^{2^{2^{\cdots}}}$ or 2 to the 2 to the 2... to the x^{th} power) are all elementary. The elementary functions have many alternative characterizations.

- One can use exponentiation instead of e_2.

- One can add cutoff subtraction and exponentiation and trade bounded recursion for bounded minimization.
- Or one can add addition and cutoff subtraction and replace bounded recursion with bounded sum and product.

Exercise 4.2 *Show that the elementary functions are closed under bounded simultaneous recursion. Hint: first show that the exponential function 2^x is in E. Then use this to show that the Gödel coding and decoding is elementary by checking all the functions involved in its derivation tree. The crux of this argument is to show by induction that addition and multiplication are bounded by compositions of exponentiation. Then use the coding to obtain an elementary reduction to bounded recursion.*

4.5 Bounding functions in the Grzegorczyk classes

We will now construct bounds on the functions in the classes E_n. The first result says that functions in E_0 can't achieve more than to add a fixed amount to a given input. This makes sense, because compositions of successor and projection can only add a fixed amount to some argument and bounded recursion can't grow faster than that.

There is a weakness in this approach to negative results: characteristic functions that run way beyond primitive recursive in complexity don't grow at all. For example, $diag(x) = \overline{sg}(f^1_x(x))$. This is a non-primitive recursive function since it differs from each unary primitive recursive function in at least one place. But it doesn't grow at all. Growth is only a symptom of complexity.

Proposition 4.3 *Each f in E_0 satisfies for n-ary $\vec{x}$:*

$$\exists_{i \leq n}\ \exists k\ \forall \vec{x}\ (f(\vec{x}) \leq x_i + k)$$

Proof. By induction on depth of E_0 derivation trees.
Base case:

$$\begin{aligned} z(x) &\leq x \\ s(x) &\leq x+1 \\ p^i_k(\vec{x}) &\leq x_i \end{aligned}$$

Inductive Case:
Suppose $h = R(g, f)$, h is bounded by e, and that g, f, e are in E_0. Then by the induction hypothesis, e satisfies the proposition. So h does as well (let i and k be the same as for e).

Suppose $h = C(f, g_1, \ldots, g_k)$, and that $f, g_1, \ldots, g_k$ are in E_0. By the inductive hypothesis, for each $i < k$,

$$\exists\ j(i)\ \forall \vec{y}\ (g_i(\vec{y}) \leq y_{j(i)} + b_i)$$

$$\exists m\ \forall \vec{x}\ (f(\vec{x}) \leq x_m + c)$$

Then for each y,

$$\begin{aligned} f(g_1(y), \ldots, g_k(y)) &< g_m(y) + c \\ &< y_{j(m)} + b_m + c \\ &= y_{j(m)} + d \end{aligned}$$

.
⊣

Notice that in the preceding proof, the only interesting case is composition. That is because only composition is allowed to build growth rates, since primitive recursion is bounded. The next result says that each function in E_1 is everywhere bounded by a linear function of the same arguments, which is also not surprising, since compositions of additions yield linear functions.

Proposition 4.4 *Each $f \in E_1$ satisfies:*

$$\exists\ _{\textit{linear } h}\ \forall \vec{x}\ (f(\vec{x}) \leq h(\vec{x}))$$

Proof. By induction on depth of E_1 derivation trees. ⊣

Exercise 4.3 *Prove it. Don't forget to add e_0 to the base case!*

The next result says that each function in E_2 is bounded by a polynomial function. That's not surprising either since compositions of linear functions and squared functions yield polynomial functions.

Proposition 4.5 *Each f in E_2 satisfies:*

$$\exists\ _{\textit{polynomial } p}\ \forall \vec{x}\ (f(\vec{x}) \leq p(\vec{x}))$$

Proof. By induction on depth of E_2 derivation trees. ⊣

Exercise 4.4 *Prove it.*

Finally, each of the successive classes E_{n+3} is bounded by iterates of its generating function e_{n+1}.

Proposition 4.6 *Each f in E_{n+3} satisfies:*

$$\forall x_1, \ldots, x_n\ \exists k\ (f(x_1, \ldots, x_n) \leq (e_{n+2})^k(\max(x_1, \ldots, x_n)))$$

Proof. By induction on depth of E_{n+3} derivation trees. Recall:

$$\begin{aligned} e_0(x,y) &= x+y \\ e_1(x) &= x^2+2 \\ e_{n+2}(0) &= 2 \\ e_{n+2}(x+1) &= e_{n+1}(e_{n+2}(x)) \end{aligned}$$

Base case: Observe that

$$\begin{aligned} e_2(0) &= 2 \\ e_2(x+1) &= e_1(e_2(x)) \\ &= e_2(x)^2+2 \end{aligned}$$

Since $e_{n+1}(x) \geq e_n(x)$, by proposition 4.2.2, it suffices in the base case to show that all the basic functions are bounded by e_2. This we do by induction on x.

$$\begin{aligned} o(x) &= 0 \\ &< 2 \\ &= e_2(0). \text{ Apply proposition 4.2.2.} \end{aligned}$$

$$\begin{aligned} s(0) &= 1 \\ &< 2 \\ &= e_2(0) \end{aligned}$$

$$\begin{aligned} s(x) = x+1 \leq e_2(x) \;\Rightarrow\; s(x+1) &= x+1+1 \\ &\leq e_2(x)+1 \\ &< e_2(x)^2+2 \\ &= e_2(x) \\ &\leq e_2(x+1) \text{ by proposition 4.2.2.} \end{aligned}$$

$$\begin{aligned} \max(\vec{x}) = 0 \;\Rightarrow\; p^i_k(\vec{x}) &= 0 \\ &< 2 \\ &= e_2(\max(\vec{x})) \end{aligned}$$

$$\begin{aligned} \max(\vec{x}) = n \wedge \max(\vec{y}) = n+1 \wedge p^i_k(\vec{x}) \leq e_2(n) \;\Rightarrow\; p^i_k(\vec{y}) &\leq n+1 \\ &\leq e_2(n)+1 \\ &< e_2(n)^2+2 \\ &= e_2(n+1) \\ &= e_2(\max(\vec{y})) \end{aligned}$$

Inductive case: we now deal with functions in E_{n+3} that are built up by C or R. Suppose $h = R(g,f)$, h is bounded by e, and that g, f, e are in E_{n+3}. Then by the induction hypothesis, e satisfies the proposition. So h does as well (let i and k be the same as for e).

Suppose $h = C(f, g_1, \ldots, g_k)$, and that $f, g_1, \ldots, g_k$ are in E_{n+3}. By the induction hypothesis, for each $i \leq k$,

$$\exists\, j(i)\; \forall \vec{y}\; (g_i(\vec{y}) \leq (e_{n+2})(\max(\vec{y}))^{j(i)})$$

$$\exists m\; \forall \vec{x}\; ((e_{n+2})(\max(\vec{x}))^m)$$

Then for each y,

$$\begin{aligned} f(g_1(\vec{y}),\ldots,g_k(\vec{y})) &< (e_{n+2})^m(\max(g(\vec{y})) \text{ by second hypothesis.} \\ &< (e_{n+2})^m(e_{n+2})^{\max(j(1),\ldots,j(k))}(\max(\vec{y})) \\ &\quad \text{by first hypothesis and proposition 4.2.2.} \\ &< (e_{n+2})^{m+\max(j(1),\ldots,j(k))} \end{aligned}$$ ⊣

Proposition 4.7 *If $f \in E_n$ then the iteration $f^y \in E_{n+1}$.*

Proof. The iteration of addition of a constant is bounded by a linear function, yielding the case for $n = 0$. Similarly, the iteration of a linear function is a polynomial function, yielding the $n = 1$ case. Let $f \in E_{n+1}$. We will consider the case of binary f, with iteration on x. When $n > 1$, we have by proposition 4.6 above, there is an m such that

$$f(x,y) \leq (e_{n-1})^m(\max(x,y))$$

Now by induction, we show:

$$f^z(x,y) \leq (e_{n-1})^{mz}(\max(x,y)) \quad (\star) \tag{4.5}$$

Base case:

$$\begin{aligned} f^0(x,y) &= x \\ &\leq \max(x,y) \\ &= (e_{n-1})^0(\max(x,y)) \end{aligned}$$

Inductive case: Now suppose that for all x, y,

$$f^z(x,y) < (e_{n-1})^{mz}(\max(x,y))$$

Then:

$$\begin{aligned} f^{z+1}(x,y) &= f^z(f(x,y),y) \text{ by the definition of iteration.} \\ &\leq f^z((e_{n-1})^m(\max(x,y)),y) \text{ by IH \& prop. 4.2.2.} \\ &\leq (e_{n-1})^{mz}(\max((e_{n-1})^m(\max(x,y)),y)) \text{ by IH.} \\ &\leq (e_{n-1})^{m(z+1)}(\max(x,y)) \text{ by prop. 4.2.2.} \end{aligned}$$

That completes the induction. Observe, further, that

$$(e_{n-1})^{mz}(\max(x,y)) \leq (e_n)(\max(x,y)+mz) \text{ by prop. 4.2.4.} \quad (\star\star). \tag{4.6}$$

So by chaining $(\star)$ with $(\star\star)$, we obtain

$$f^z(x,y) \leq (e_n)(\max(x,y)+mz) \quad (\star\star\star) \tag{4.7}$$

Now the iteration f^z is defined by primitive recursion in terms of f.
By $(\star\star\star)$, this primitive recursion is bounded by $e_n \in E_{n+1}$.
So $f^z \in E_{n+1}$. ⊣

4.6 Hierarchy Theorem

Proposition 4.8 *E_n is a subset of E_{n+1}.*

Proof sketch. By proposition 4.2.3, each generating function for a class is in all the higher classes. Then all the closure conditions of the lower class are satisfied by the higher class. ⊣

Proposition 4.9 *$e_n \in E_{n+1} - E_n$.*

Proof sketch. Recall:

$$\begin{aligned} e_0(x,y) &= x+y \\ e_1(x) &= x^2+2 \\ e_{n+2}(0) &= 2 \\ e_{n+2}(x+1) &= e_{n+1}(e_{n+2}(x)) \end{aligned}$$

The membership claims are all true by definition of E_n. The non-memberships are established by induction on n.
Base cases:

- By proposition 4.3, $e_0 \notin E_0$, since there is no constant bound on e_0.
- By proposition 4.4, $e_1 \notin E_1$, since e_1 is not a linear function of x.
- By proposition 4.5, $e_2 \notin E_2$, since e_2 is not polynomial (its calculation involves arbitrarily high exponents).

Inductive case: Suppose that $e_{n+2} \notin E_{n+2}$. Then by proposition 4.6,

$$\forall k\ \exists x\ (e_{n+2}(x) > (e_{n+1})^k(x)$$

Now we show by induction on k that:

$$\forall k\ \exists x\ (e_{n+3}(x) > (e_{n+2})^k(x)$$

Base Case: For $k = 0$, we have

$$\begin{aligned} e_{n+3}(0) &= 2 \\ &> 0 \\ &= (e_{n+2})^0(0) \end{aligned}$$

Inductive Case: Now suppose that for some x

$$e_{n+3}(x) > (e_{n+2})^k(x)$$

Then

$$\begin{aligned} e_{n+3}(x+1) &= e_{n+2}(e_{n+3}(x)) \\ &> e_{n+2}((e_{n+2})^{k'}(x)) \\ &= (e_{n+2})^{k'+1}(x) \end{aligned}$$

Observe that the inequality is by proposition 4.2.2. ⊣

Proposition 4.10 $\bigcup_i E_i = Prim.$

Proof. The basic functions and composition are no problem because each E_n has them. But none of the classes has primitive recursion, so we must show that somehow the effect of primitive recursion results from taking the union of the classes. So why would we think it's true? Because the successive generating functions provide increasing bounds so that each unbounded recursion becomes bounded by some level. By proposition 4.7, the iteration f^y of a function f in E_n is in E_{n+1}. Then one proves (nontrivial) that all unary primitive recursive functions can be defined by iteration and composition over some simple primitive recursive functions (cf. Rose pp. 17-22). The idea, as usual, is to use an encoding to allow the iteration to simulate unary primitive recursion. Then one reduces n-ary primitive recursion to unary primitive recursion.

An easier way to prove this proposition is to do it directly by induction on derivation complexity, bounding compositions and primitive recursions over functions in one class by compositions of higher generating functions. This is more direct, but then one doesn't end up with nice results like proposition 4.8 along the way because the bounds are loose (e.g., jumping by 3). $\dashv$

Proposition 4.11 *The Péter function is not primitive recursive.*

Proof: The Péter function p outgrows each e_n. Thus, by proposition 4.9, p is not in any class E_n. By proposition 4.10, p is not primitive recursive. $\dashv$

4.7 Onward and Upward

If you like this kind of thing, the Grzegorczyk hierarchy can be extended by recursion on infinite ordinals to obtain classes corresponding to double recursion, triple recursion, etc. See Rose for a systematic presentation. We are now going to leave the world of pure recursion for the more elegant theory of the partial recursive functions.