Phil 320

Chapter 7: Recursive sets and relations

(**Note:** We cover only section 7.1.)

0. Introduction

Significance of and main objectives for chapter 7:

1. Chapter 7 generalizes the notion of recursive (or primitive recursive) functions. Not just functions, but *sets* and *relations*, can be recursive (or p.r.). You should:

a) understand what it means for a relation to be recursive (p.r.)
b) learn some basic techniques for showing that relations are recursive (p.r.)

2. Provides some new and powerful techniques for showing that functions are recursive or p.r. You should learn these techniques. The three most important ones:

a) definition by cases
b) bounded minimization and maximization
c) minimization/maximization with p.r. bound

3. Why learn so much about primitive recursive and recursive functions? We use these techniques in chapter 8 (every Turing-computable function is recursive) and chapter 11 (there is no effective procedure to decide on logical implication). They are crucial in chapters 15-18 (Godel's Incompleteness Theorem).

I. Recursive sets and relations

1. Notation: sets

$x \in$ S, Sx, S(x) all used to signify that x is in the set S

2. Notation: two-place relations

A two-place relation *R* on natural numbers is (formally) a set of ordered pairs of natural numbers.

$(x, y) \in R$, Rxy, $R(x, y)$, xRy all signify that the relation R holds between x and y

Examples: $x < y$ (less-than), $x = y$ (identity), $x \geq y$ (greater-than-or-equal-to)

3. Notation: n-place relations

An n-place relation *R* on natural numbers is a set of n-tuples of natural numbers. We signify that the relation R holds among $x_1,\ldots, x_n$ in the following ways:

$x_1,\ldots, x_n \in$ R, R $x_1,\ldots, x_n$, R$(x_1,\ldots, x_n)$

Example: $R(x_1, x_2, x_3) \leftrightarrow x_1 + x_2 = x_3$. $\leftrightarrow$ means 'if and only if'

For convenience, write **x** for the n-tuple $(x_1,\ldots, x_n)$, and use $\mathbf{x} \in$ R or R**x**.

4. Characteristic functions

If *S* is any set, then its characteristic function c_S is defined by

$c_S(x) = \{1, x \in S$
$\qquad\quad \{0, x \notin S.$

If R is an n-place relation, its characteristic function is

$c_R(x_1,\ldots, x_n) = \{1 \text{ if } R(x_1,\ldots, x_n)$ OR $c_R(\mathbf{x}) = \{1 \text{ if } R\mathbf{x}$
$\qquad\qquad\qquad \{0 \text{ if not } R(x_1,\ldots, x_n)$ $\qquad \{0 \text{ if not } R\mathbf{x}$

Each n-place relation corresponds to a unique n-place characteristic function.

***5. Three sorts of decidability: effective, recursive and primitive recursive**

Decidability is to relations (or sets) as *computability* is to functions. We'll define analogues that correspond to effectively computable, recursive and p.r. functions. The key idea is to make use of characteristic functions.

a) A set *S* of natural numbers is *effectively decidable* if there is an effective procedure that, applied to any number *x*, will halt with a correct answer as to whether *x* belongs to *S*.

A relation *R* is *effectively decidable* if there is an effective procedure that, applied to a n-tuple **x** of numbers, will halt with a correct answer as to whether **x** belongs to *R*.

Observe: *R* is *effectively decidable* if and only if c_R is effectively computable.

b) *S* is *recursively decidable* (or just *recursive*) if its characteristic function is recursive.

R is *recursively decidable* (or just *recursive*) if its characteristic function is recursive.

c) *S* is *primitive recursive* if c_S is p.r.; *R* is *primitive recursive* if c_R is p.r.

Church's Thesis implies that every effectively decidable relation is recursive.

6. Recursive Relations 101 (basic technique)

To show R is recursive (p.r.), show that its characteristic function c_R is recursive (p.r.).

Examples:

1) The identity relation $x = y$ is p.r.

Proof: Write $c_=$ for the characteristic function. Then $c_=(x, y) = \overline{sg}(|x - y|)$.

2) The non-identity relation $x \neq y$ is p.r.

Proof: $c_{\neq}(x, y) = sg(x \dot{-} y) + sg(y \dot{-} x)$; or, $c_{\neq}(x, y) = sg(|x - y|)$.

3) The relation $x < y$ is p.r.

Proof: $c_<(x, y) = sg(y \dot{-} x)$ [Clearly, $y < x$ is also p.r.]

4) The relation $x \leq y$ is p.r.

Proof: $c_{\leq}(x, y) = 1 \dot{-} sg(x \dot{-} y)$; or using Ex. 3), $c_{\leq}(x, y) = 1 - c_<(y, x)$.

7. Recursive Relations 201 (advanced techniques)

***Technique i): Substitution**

Example 1. $R^*(x, y) \leftrightarrow x^2 \leq y$ is a p.r. relation.

Proof: $R(x, y) \leftrightarrow x \leq y$ is a p.r. relation. $f(x) = x^2$ is a p.r. function. R* is obtained by substituting f(x) for the first term of the relation R. So R* is a p.r. relation.

Example 2. $R^*(x, y) \leftrightarrow x \cdot y^2 \leq 5$ is a p.r. relation

Proof: $f(x, y) = x \cdot y^2$ and $g(x, y) = 5$ are p.r. relations. R* is obtained by substituting f and g for the two terms of the relation R (of example 1). So R* is a p.r. relation.

Definition. Given a relation $R(y_1, \ldots, y_m)$ and m total n-place functions $f_1(x_1,\ldots, x_n), \ldots, f_m(x_1,\ldots, x_n)$, the n-place relation R* obtained by substitution of the f_i into R is:

$$R^*(x_1,\ldots, x_n) \leftrightarrow R(f_1(x_1,\ldots, x_n), \ldots, f_m(x_1,\ldots, x_n))$$

Substitution of recursive (p.r.) functions into R yields a recursive (p.r.) relation R*.

Proof: if c* is the characteristic function of R* and c for the characteristic function of R, then

$$c^*(x_1,\ldots, x_n) = c(f_1(x_1,\ldots, x_n), \ldots, f_m(x_1,\ldots, x_n)).$$

So if the f_i's and c are recursive, so is c*; and if all are p.r., then so is c*.

*Technique ii): applying logical operations

We can define new relations from old ones using basic logical operations like 'not' and 'and'. If we start with a recursive (p.r.) relation, we end with a recursive (p.r.) relation.

Example: $R(x, y, z) \leftrightarrow [\sim(x^y \leq z) \vee (2z+y > x)] \,\&\, (x \cdot z = y)$

R is p.r. because R is put together from p.r. relations using logical operations.

a) *Negation:* $R^*(x_1,\ldots, x_n) \leftrightarrow \sim R(x_1,\ldots, x_n)$.

R* is p.r. (recursive) if R is: $c_{R^*} = 1 \dot{-} c_R$ **Ex:** $R^*(x, y) \leftrightarrow \sim(x < y)$

b) *Conjunction:*

$R^*(x_1,\ldots, x_n) \leftrightarrow R_1(x_1,\ldots, x_n) \,\&\, R_2(x_1,\ldots, x_n)$ **Ex:** $x < y \,\&\, y < x^2$

R* is p.r. (recursive) if R_1 and R_2 are: $c_{R^*} = c_1 \cdot c_2$; or $c_{R^*} = \min(c_1, c_2)$.

c) *Disjunction:*

$R^*(x_1,\ldots, x_n) \leftrightarrow R_1(x_1,\ldots, x_n) \vee R_2(x_1,\ldots, x_n)$ **Ex:** $x < y \vee x < z$

R* is p.r. (recursive) if R_1 and R_2 are: $c_{R^*} = \max(c_1, c_2)$ [or: $sg(c_1 + c_2)$].

A conjunction/disjunction of any finite number of p.r. relations is a p.r. relation.

d) *Bounded universal quantification:*

If $R(x_1,\ldots, x_n, y)$ is an (n+1)-place relation, we define an associated relation:

$R^*(x_1,\ldots, x_n, u) \leftrightarrow \forall y < u\, R(x_1,\ldots, x_n, y)$
$[R^*(\mathbf{x}, u) \leftrightarrow R(\mathbf{x}, 0) \,\&\, R(\mathbf{x}, 1) \,\&\, \ldots \,\&\, R(\mathbf{x}, u\text{-}1)]$

Similarly, we can define

$R^*(x_1,\ldots, x_n, u) \leftrightarrow \forall y \leq u\, R(x_1,\ldots, x_n, y)$

R* is p.r. (recursive) if R is:

$C_{R^*}(\mathbf{x}, u) = \prod_{y=0}^{u-1} c(\mathbf{x}, y)$ (read $u \dot{-} 1$ for u-1).

For this number is 1 only if each of $R(\mathbf{x}, 0)$, $R(\mathbf{x}, 1)$, …, $R(\mathbf{x}, u\text{-}1)$ hold. (Add $R(\mathbf{x}, u)$ for the second version.)

e) *Bounded existential quantification:*

If $R(x_1,\ldots, x_n, y)$ is an (n+1)-place relation, we define

$R^*(x_1,\ldots, x_n, u) \leftrightarrow \exists y < u\, R(x_1,\ldots, x_n, y)$

$[R^*(\mathbf{x}, u) \leftrightarrow R(\mathbf{x}, 0) \vee R(\mathbf{x}, 1) \vee \ldots \vee R(\mathbf{x}, u\text{-}1)]$

Similarly, $R^*(x_1,\ldots, x_n, u) \leftrightarrow \exists y \leq u\, R(x_1,\ldots, x_n, y)$

To see that R* is p.r. (recursive) if R is:

$c_S(\mathbf{x},u)=sg(\sum_{y=0}^{u-1} c(\mathbf{x}, y))$ (read $u \dot{-} 1$ for u-1). (Add $c(\mathbf{x}, u)$ for the second case).

Example 1: Prime numbers.

Px ↔ x is prime is a p.r. set.

Proof: $Px \leftrightarrow 1 < x \,\&\, \forall u < x\, \forall v < x\, (u \cdot v \neq x)$.

$1 < x$ is clearly p.r. $u \cdot v \neq x$ is p.r. by substitution. Two applications of bounded universal quantification mean the second conjunct is p.r. By conjunction, P is p.r.

Example 2: Perfect squares.

$Sx \leftrightarrow \exists y \leq x\ (x = y^2)$ is p.r.

Proof: $x = y^2$ is p.r. by substitution. Then S is obtained by bounded existential quantification.

II. New ways to show that a function is p.r.

*1. Definition by cases

Example: $f(x, y) = \max(x, y) = \begin{cases} x, & x \geq y \\ y, & y > x \end{cases}$

The function f is defined differently for different *cases*. The cases are mutually exclusive (no overlap) and exhaustive (exactly one must hold). The relation defining each case is p.r., and the function defined for each case is p.r. So f is p.r.

Formally:
Suppose $f(x, y) = \begin{cases} g_1(x, y) \text{ if } C_1(x, y) \\ \ldots \\ g_n(x, y) \text{ if } C_n(x, y) \end{cases}$ where C_i's are exclusive and exhaustive.

If each g_i and each C_i is recursive, then f is recursive.
If each g_i and each C_i is p.r., then f is p.r.

Proof: Let c_i be the characteristic function of C_i. If each c_i is recursive (p.r.), then

$$f(x, y) = g_1(x, y)c_1(x, y) + \ldots + g_n(x, y)c_n(x, y)$$

is recursive (p.r.).

Hint: Use 'circling' technique. Circle each g_i and each C_i, and explain why the non-obvious ones are p.r. (or recursive). That's it!

Example 2: $f(x, y) = \begin{cases} x \dot{-} y, & x \geq y \\ y \dot{-} x, & x < y \end{cases} = |x - y|$

f is p.r., by cases.

2. Bounded maximization and minimization

If R is recursive (p.r.) and w is a fixed number (the upper bound for the search), set

$\text{Min}[R](\mathbf{x}, w) = \begin{cases} \text{smallest } y \leq w \text{ for which } R(\mathbf{x}, y) & \text{if such a y exists} \\ w+1 & \text{if no such y exists} \end{cases}$

and

$\text{Max}[R](\mathbf{x}, w) = \begin{cases} \text{largest } y \leq w \text{ for which } R(\mathbf{x}, y) & \text{if such a y exists} \\ 0 & \text{if no such y exists} \end{cases}$

Min[R] and Max[R] are recursive (p.r.) total functions. (**Note:** Each process defines an n-place *function* based on an (n+1)-place relation R and a fixed bound, w.)

Proof: Let c be the characteristic function for ~R. Then

$$\text{Min}[R](\mathbf{x}, w) = \sum_{y=0}^{w} \prod_{i=0}^{y} c(\mathbf{x}, i) \qquad \text{(since the product is 1 for } w \text{ up to y, and 0 after)}$$

Similarly, if c is the characteristic function for R, then

$$\text{Max}[R](\mathbf{x}, w) = \sum_{y=0}^{w} sg \sum_{i=y+1}^{w} c(\mathbf{x}, i) .$$

Example: *Quotients and remainders*

For any x and y with y > 0, we can write $x = q \cdot y + r$, where $r < y$.

Examples:	$12 = 1 \cdot 7 + 5$	quo(12, 7) = 1; rem(12, 7) = 5
	$43 = 8 \cdot 5 + 3$	quo(43, 8) = 5; rem(43, 8) = 3

q is called the *quotient* quo(x, y). [set quo(x, 0) = 0]
r is called the *remainder* rem(x, y). [set rem(x, 0) = x]

quo is p.r.:

quo(x, y) = {largest $q \leq x$ such that $q \cdot y \leq x$ if $y \neq 0$
{0 if y = 0

Note that quo is defined by cases AND bounded max.

rem is p.r.:

$\text{rem}(x, y) = x \dot{-} [\text{quo}(x, y) \cdot y]$

The relation R(x, y) ↔ x divides y evenly is p.r.

Proof: R(x, y) ↔ rem(y, x) = 0.

*3. Minimization with primitive recursive bound.

Example: $f(x,y) = x^3 \dot{-} y^2$. Recall that Mn[f] is a recursive function.

We can see that Mn[f] is a total function. Each value of Mn[f] seems to be computable in finitely many steps. Is Mn[f] primitive recursive?

Yes! Use **minimization with primitive recursive bound**. Replace the fixed upper bound w with an upper bound g(x) that can vary, but is a p.r. function of x.

Theorem: Suppose *f* is a regular primitive recursive (n+1)-place function, and there is a p.r. n-place function g such that for every $\mathbf{x} = (x_1,\ldots, x_n)$,

The least y such that $f(\mathbf{x}, y) = 0$ is always $\leq g(\mathbf{x})$. [Need only search $0,\ldots, g(\mathbf{x})$.]

Then Mn[f] is not only recursive but also primitive recursive.

Proof:

$\text{Mn}[f](\mathbf{x}) = \text{Min}[R](\mathbf{x}, g(\mathbf{x}))$ where $R(\mathbf{x}, y) \leftrightarrow f(\mathbf{x}, y) = 0$ is a p.r. relation;

substituting in $g(\mathbf{x})$ for w preserves p.r.

A very useful technique. (Go back to $f(x,y) = x^3 \dot{-} y^2$.)

Related technique: Minimization with recursive relations.

Let R be an (n+1)-place recursive relation. Define the function

$r(x_1,\ldots, x_n)$ = least y such that $R(x_1,\ldots, x_n,y)$

Then r is recursive.

Proof: If c is the characteristic function of ~R, then $c(\mathbf{x}, y) = 1$ right up to the least y such that $R(\mathbf{x}, y)$, and that is the first y where $c(\mathbf{x}, y) = 0$. So

$r(x_1,\ldots, x_n) = \text{Mn}[c](x_1,\ldots, x_n)$.

4. Final examples of primitive recursive functions

Example 1: Next prime number. f(x) = least y such that $x < y$ and y is prime. [Since we can find such a $y \leq x! + 1$, we have a p.r. bound.]

Example 2: Two kinds of logarithm.

lo(x, y) = {largest z such that y^z divides x evenly if $x,y > 1$
{0 otherwise

Examples: lo(3,6) = 0, lo(6,3) = 1, lo(24,2) = 3, lo(56,2) = 3

The cases are p.r., and the one tricky function ('largest z…') is defined via minimization with p.r. bound (since z will be less than x). So lo(x, y) is p.r.

lg(x, y) = {largest z such that $y^z \leq x$ if $x,y > 1$
{0 otherwise

Examples: lg(3, 6) = 0, lg(6,3) = 1, lg(24, 2) = 4, lg(56, 2) = 5

Again, the cases are p.r. and the tricky function ('largest z…') is defined via minimization with p.r. bound (since z will be less than x). So lg(x, y) is p.r.

Examples related to coding. Code the sequence $(a_0, a_1, a_2, \ldots a_{n-1})$ as the number $2^n 3^{a0} 5^{a1} 7^{a2} \ldots \pi(n)^{an-1}$, where π(n) is the n'th prime (counting 2 as the 0'th prime).

Example 1: The n'th prime. π(n) = the n'th prime number. π is p.r. *Proof*: π(0) = 2, and π(n′) = next prime larger than π(n).

Example 2: Length. lh(s) = *length* of the sequence $(a_0,a_1,\ldots a_{n-1})$ coded by s, if s codes any sequence (0 otherwise). lh is p.r. *Proof*: lh(s) = lo(s, 2). The new coding stores the length at the start.

Example 3: Entries. If s codes $(a_0, a_1, \ldots, a_{n-1})$, then Ent(s, i) = a_i (and otherwise 0). Ent is p.r. *Proof*: Ent(s, i) = lo(s, π(i+1))

What strategy should I use to show a function f is p.r.?

1) If you can see a function you know to be p.r. embedded in the function of interest, use composition.

Ex: $f(x, y, z) = x^{y+z} \cdot z^{y \dot{-} x}$

2) Otherwise, the most common are primitive recursion and bounded minimization or maximization, possibly in combination with definition by cases. If it's not obvious what to do, it's probably one of these.

To use p.r.:

a) Write out a table of values.

b) If it looks like there's a simple rule for getting from f(x, y) to f(x, y+1) for each x [in 2-place case], it's probably p.r. Inspecting the table gives you the clue to what the cases are, and how to define next case in terms of previous.

Example: $f(x) = [\sqrt{x}]$ = greatest integer $\leq \sqrt{x}$.

Example: $f(x, y) = [y\sqrt{x}]$.

To use bounded Mn (or bounded max):

First write out an informal description, "the least m such that", or "the largest m such that". Then you have to define a function that makes m the final argument, and apply bounded max/min.

Example: $f(x) = [\sqrt{x}]$ = largest m with $m^2 \leq x$. This is p.r. by bounded max: an upper bound for m is just x, which is p.r..