CS109A Notes for Lecture 3/8/95

Properties of Binary Relations

1. *Symmetry*: aRb implies bRa.

Example: Define aR_1b iff $a + b$ is divisible by 3. R_1 is symmetric.

Example: The empty relation is symmetric.

- Remember, any statement "A implies B" is true when A is false.

2. *Transitivity*: aRb and bRc imply aRc.

Example:

- $<$ on integers is transitive.
- So is the empty relation.
- R_1 is not transitive. e.g., $2R_11$ and $1R_15$, but $2R_15$ is false.
    - □ Note: a single counterexample proves a relation *doesn't* have a certain property, but a general proof is needed to show it does.

3. *Reflexivity*: aRa for all a in the (declared) domain of R.

Example:

- $\leq$ on integers is reflexive.
- $<$ is not.
- The empty relation is not reflexive unless the *declared* domain is empty.

4. *Antisymmetry*: aRb and bRa imply $a = b$.

Example:

- $\leq$ and $<$ on integers are both antisymmetric.
- R_1 is not; e.g., $1R_12$ and $2R_11$.

5. *Comparability*: For any a and b in the declared domain of R, at least one of aRb and bRa holds.

Example:

- $\leq$ on integers is comparable.
- $<$ is not, because of the possibility $a = b$.
- R_1 is not; e.g., neither $2R_13$ nor $3R_12$.

Partial Orders

A relation that is transitive and antisymmetric.

Example: $\leq$ or $<$ on integers.

Example: The subsets of a given set A form a partial order.

- Transitivity: If $B \subseteq C$ and $C \subseteq D$, then $B \subseteq D$.
- Antisymmetry: If $B \subseteq C$ and $C \subseteq B$, then $B = C$.

Example: C = "component of" on auto parts, e.g. $tireCwheel$, $nutCwheel$, $wheelCcar$, $nutCengine$, $pistonCengine$.

Total Orders

Comparable partial order.

Example:

- $\leq$ or $<$ on integers.
- Not $\subseteq$ on subsets of A, as long as A has at least two members.
  - ☐ e.g., if $A = \{0, 1\}$, neither $\{0\} \subseteq \{1\}$ nor $\{1\} \subseteq \{0\}$ is true.
- Not "component of."
  - ☐ For example, neither $wheelCengine$ nor $engineCwheel$ are true.

Equivalence Relations

Reflexive, symmetric, transitive.

Example: Common example: *congruence modulo m*.

- i.e., iEj iff i and j have the same remainder when divided by m.

- Be careful how remainders are computed for negative numbers. The remainder is how much must be subtracted from i to reach a multiple of m.

  - □ e.g., $-5 \bmod 3 = 1$, although $5 \bmod 3 = 2$.

Equivalence Classes

If E is an equivalence relation, we can partition the domain of E into sets called *equivalence classes* such that:

- aEb if and only if a and b are in the same equivalence class.

- Proof on p. 393 FCS that this definition makes sense, i.e., it is possible to partition the domain of an equivalence relation in this way.

Example: If E is congruence modulo m, the equivalence classes are the m sets of integers with common remainders, e.g., $\{0, m, 2m, \ldots\}$, $\{1, m+1, 2m+1, \ldots\}$, etc.

- Each set also includes negative integers.

Example: *Balanced parenthesis strings* can be defined as those strings of parens that

1. Have an equal number of left and right parens.

2. No prefix has more right parens than left.

- Good model of problem in compiling: Scan a string of parens left-to-right and determine whether it is balanced.

  - □ Equivalence-relation question: how much do we have to remember about the string as we scan it?

- Define sEt if strings s and t have the property that for all strings x, sx is balanced iff tx is balanced.

- ☐ i.e., all we have to remember about the string is what equivalence class it belongs in.

- Easy to check E is an equivalence relation, e.g., transitivity: "sx is balanced iff tx is balanced" and "tx is balanced iff rx is balanced" imply "sx is balanced iff rx is balanced."

- What are equivalence classes?

  1. There is one class of "dead" strings. they have had a point with more right parens than left, so no continuation can lead to a balanced string.

  2. For each i there is a class C_i of strings with i more left parens, and no prefix whose right parens exceed the left.

- If $i \neq j$, then choosing $x =))\cdots)$ (i parens) leads to balance for any string in C_i, but no string in C_j.

  - ☐ Thus, strings in different classes cannot be equivalent.

- If s and t are both in C_i, and x is a string such that sx is balanced, then tx is also balanced. Why?

  - ☐ Thus, all strings in the same class are equivalent.

- Conclusion: it is sufficient, when recognizing balanced strings, to record:

  a) Has the difference of left-parens minus right-parens ever gone negative?

  b) If not, what is the current difference?