CS 145 – Foundations of Computer Science

Professor Eric Aaron

Lecture – W F 1:30pm
Lab – F 3:30pm

Lecture Meeting Location: SP 105
Lab Meeting Location: SP 309

Business

• HW4 out now, due Oct. 26 / Oct. 27 (as always, see assignment sheet for exact deadlines)
  – Extension on HW4? (Email me your votes?)

• Reading: Ch.4.1-4.6
  – Our coverage of the material will be different from that in the textbook, but it’s good to see the textbook’s presentation, as well

• Reading: Prof. Hunsberger’s document “The Natural Numbers, Induction, and Numeric Recursion”
  – Posted on the Additional Notes / Readings page of the CS145 website
Business, pt. 2

- Visiting Speaker next week (Wed., Oct. 25)!
- Presentation about the AIT-Budapest study abroad program / JYA option for Computer Science
  - Title: CS Study Abroad in Budapest
  - Speaker: Gábor Bojár, Founder and Professor of IT Entrepreneurship, AIT-Budapest program
  - Date: Wednesday, Oct. 25
  - Place: SP 105
  - Time: 3:15pm
  - Snacks / Refreshments: Will be served!

A General Note about Proving Stuff

- As we’ve discussed in class, proofs are important, and understanding how to structure a proof is important

**Generally Good Idea:** When proving a claim about something, let the definition of that thing guide your proof.

- For anything defined by recursion, the preferred proof technique is typically induction.
- This is because of the same Generally Good Idea that guides our proofs of transitivity, set equality, etc.—follow the relevant definitions!
  - (But we’ve proved things about numbers by induction… hm…)

**Important Observation:** When we prove something by induction, we are typically being guided to do so by an underlying recursive definition.
Recursive (or Inductive) Definition of a Set

• A recursive definition of a set S consists of three components:
  – **Base**: One or more foundational elements of S
  – **Induction**: One or more rules to construct new elements of S from existing elements of S
  – **Closure**: The condition that S consists of *all and exactly* the elements derived from the base elements and induction rules.
    (In the context of a definition, this is often assumed rather than explicitly stated—the fact that it is a *definition* means that the set is exactly the elements thus specified.)

• What are some recursively defined sets we’ve already seen, and what are their definitions?

A Recursively Defined Set: Propositional Logic Expressions

• Recall our expressions for propositional logic: *propositional letters* (i.e., variables), negations, conjunctions, etc.
• A recursive definition of the set of all propositional logic expressions:
  – **Base**: Given an initial set A of propositional letters (e.g., p, q, r, …), all elements of A are propositional logic expressions
  – **Induction**: If P, Q are propositional logic expressions, then the following are also propositional logic expressions (note that the parentheses are part of the expressions)
    • (¬P)
    • (P ^ Q)
    • (P v Q)
    • (P → Q)
    • (P ↔ Q)
  – (implicit, when unstated) **Closure**: The set of propositional logic expressions is all and only these expressions

(See Makinson, pg. 96)

Two Very Important Questions:
1. Is this a *good* recursive definition for the set of propositional logic expressions?
2. What are the trade-offs for such a definition?
A Recursively Defined Set: Propositional Logic Expressions

- Recall our expressions for propositional logic: propositional letters (i.e., variables), negations, conjunctions, etc.
- A recursive definition of the set of all propositional logic expressions:
  - Base: Given an initial set $A$ of propositional letters (e.g., $p, q, r, \ldots$), all elements of $A$ are propositional logic expressions
  - Induction: If $P, Q$ are propositional logic expressions, then the following are also propositional logic expressions (note that the parentheses are part of the expressions)
    - $(\neg P)$
    - $(P \land Q)$
    - $(P \lor Q)$
  - (implicit, when unstated) Closure: The set of propositional logic expressions is all and only these expressions

(See Makinson, pg. 96)

Two Very Important Questions:
1. Is this a better recursive definition for the set of propositional logic expressions? (Is it equivalent to the previous one?)
2. What are the trade-offs for such a definition?

A Recursively Defined Set: The Natural Numbers

- As suggested by all of our inductive proofs about numbers, there is also a recursive definition of the natural numbers
- The Peano axioms are conventionally taken as a definition of the natural numbers (here, let $N$ stand for the natural numbers):
  1. There exists a number 0 s.t. $0 \in N$
  2. Every natural number $n$ has a natural number successor, denoted by $S(n)$
  3. There is no $n$ in $N$ s.t. $S(n) = 0$
  4. Distinct natural numbers have distinct successors: if $a \neq b$, then $S(a) \neq S(b)$
  5. Let $P$ be a property of the natural numbers such that:
    - $P(0)$ holds
    - For every $a$ in $N$, if $P(a)$ holds, then $P(S(a))$ holds
  If both of those conditions are true, then $P(n)$ holds for all $n$ in $N$.

Axioms 1 and 2 give the recursive construction of the elements of $N$. The other Axioms are properties of $N$. Axiom 5 is sometimes called the Axiom of Induction.
Peano Examples

(Like, etudes?)

• The Peano axioms:
  1. There exists a number 0 s.t. 0 ∈ N
  2. Every natural number has a natural number successor, denoted by S(n)
  3. There is no n in N s.t. S(n) = 0
  4. Distinct natural numbers have distinct successors: if a ≠ b, then S(a) ≠ S(b)
  5. Let P be a property of the natural numbers such that:
     • P(0) holds
     • For every a in N, if P(a) holds, then P(S(a)) holds
   If both of those conditions are true, then P(n) holds for all n in N.

• Exercises and examples:
  – How would we write the number 2 in this notation? The number 5?
  – Could we write the number -1 in this notation?

Peano Examples

(Like, etudes?)

• The Peano axioms:
  1. There exists a number 0 s.t. 0 ∈ N
  2. Every natural number has a natural number successor, denoted by S(n)
  3. There is no n in N s.t. S(n) = 0
  4. Distinct natural numbers have distinct successors: if a ≠ b, then S(a) ≠ S(b)
  5. Let P be a property of the natural numbers such that:
     • P(0) holds
     • For every a in N, if P(a) holds, then P(S(a)) holds
   If both of those conditions are true, then P(n) holds for all n in N.

• Exercises and examples:
  – Are there two different ways to write the number 2? (Hint: Prove the following Theorem)

• Theorem: Every natural number n has one of two forms, either n=0 or n=S(m) for some m
  – Proof: By induction!

(See Hunsberger’s “The Natural Numbers...” document)