CSC236 winter 2020, week 3: structural induction, well-ordering

See section 1.2-1.3 of course notes

Colin Morris
colin@cs.toronto.edu
http://www.cs.toronto.edu/~colin/236/W20/

January 20, 2020

Outline

Principle of well-ordering

Every non-empty subset of $\mathbb{N}$ has a smallest element.

Surprisingly, turns out to be equivalent to principle of mathematical induction / complete induction. (Theorem 1.1 in Vassos course notes)

Every $n > 1$ has a prime factorization

For sake of contradiction, assume this is false. i.e.

$$S = \{n \in \mathbb{N} \mid n > 1 \land n \text{ is not the product of primes}\}$$

is non-empty. By PWO, S has a smallest element, call it j.

Every $n > 1$ has a prime factorization

For sake of contradiction, assume this is false. i.e.

$$S = \{n \in \mathbb{N} \mid n > 1 \land n \text{ is not the product of primes}\}$$

is non-empty. By PWO, S has a smallest element, call it j.
Case 1: j is prime. **Contradiction!**
Case 2: j is composite. Let $a, b \in \mathbb{N}$ such that $j = a \times b \land 1 < a < j \land 1 < b < j$ (by definition of composite).
$a, b \notin S$, since j was chosen to be the smallest element. So a and b each have a prime factorization. We can concatenate them to form a prime factorization of j.
Contradiction!
In each case, we derived a contradiction, so our premise is false. S must be empty.
$\forall n \in \mathbb{N} - \{0, 1\}$, n has a prime factorization.

Round-robin tournament cycles

Round-robin tournament $\equiv$ every player faces every other player once.
Consider "cycles" of matchups such as...

- Naomi beats Kim
- Kim beats Monica
- Monica beats Serena
- Serena beats Naomi

Claim: Any round-robin tournament having at least one cycle has a 3-cycle.

Proof: if a RR tournament has a cycle, it has a 3-cycle

Recursively defined sets

Sets defined in terms of one or more ‘simple’ examples, plus rules for generating elements from other elements.

Example:

- A single node is a complete binary tree
- If t_1 and t_2 are complete binary trees, then a new node joined to t_1 and t_2 as its children form a complete binary tree

We can use structural induction to prove properties of such sets.

Structural induction proof outline

For some recursively defined set S...

1. Define predicate with domain S
2. **Basis**: verify $P(x)$ for 'basic' element(s) $x \in S$
3. **Inductive step**: show that each rule that generates other elements of S preserves P-ness. i.e. for each rule...
   3.1 Choose arbitrary elements of S
   3.2 Assume predicate holds for those elements
   3.3 Use assumption to show that $P(z)$ holds, where z is an element generated from our previously chosen elements.

Prove: all complete binary trees have an odd number of nodes

1. Predicate

Prove: all complete binary trees have an odd number of nodes

2. Basis

Prove: all complete binary trees have an odd number of nodes

3. Inductive step

Compare with simple induction

Define $\mathbb{N}$ as the smallest[1] set such that:

1. $0 \in \mathbb{N}$
2. $n \in \mathbb{N} \implies n + 1 \in \mathbb{N}$

[1] Why is this necessary?

Strings with matching parentheses

Define $\mathcal{B}$ as the smallest set such that...

1. $\epsilon \in \mathcal{B}$ `# where` ϵ `denotes the empty string`
2. If $b \in \mathcal{B}$, then $(b) \in \mathcal{B}$
3. If $b_1, b_2 \in \mathcal{B}$, then $b_1 b_2 \in \mathcal{B}$ `# closed under concatenation`

Examples of elements?

A claim about $\mathcal{B}$

Define...

- $L(s) = \#$ of occurrences of (in s
- $R(s) = \#$ of occurrences of) in s

Claim: $\forall s \in \mathcal{B}$, if s' is a prefix of s, then $L(s') \geq R(s')$.

Prove: prefixes of strings of balanced parens are left-heavy

$P(s)$:

Basis:

Prove: prefixes of strings of balanced parens are left-heavy

Inductive step