lec1:introduction

1.Logic

Here is the syntax of the formulas of predicate logic. In this course we mainly use logical formulas to precisely express theorems.

$A \wedge B$ conjunction

$A \vee B$ disjunction

$A \Rightarrow B$ (material)implication

A iff B equivalence

$\neg A$ negation

$\forall$ x. A universal quantification

$\exists$ x. A existential quantification

After syntaxwe have semantics. The meaning of a formula is expressed in terms of truth.

- $A \wedge B$ is true iff A is true and B is true.
- $A \vee B$ is true iff A is true or B is true (or both are true).
- $A \Rightarrow B$ is true iff it is not the case that A is true and B is false.
- A iff B is true iff A and B have the same truth value.
- $\neg A$ is true iff A is false
- $\forall$ x. A is true iff A is true for all possible values of x.
- $\exists$ x. A is true iff A is true for at least one value of x.

Note the recursion: the truth value of a formula depends on the truth values of its sub-formulas. This prevents the above definition from being circular. Also, note that the apparent circularity in defining iff by using 'iff' is only apparent—it would be avoided in a completely formal definition.

Remark. The definition of implication can be a little confusing. Implication is not 'if-then-else'. Instead, you should think of $A \Rightarrow B$ as meaning 'if A is true, then B must also be true. If A is false, then it doesn't matter what B is; the value of $A \Rightarrow B$ is true'. Thus a statement such as $0 < x \Rightarrow x2 \geq 1$ is true no matter what the value of x is taken to be (supposing x is an integer). This works well with universal quantification, allowing the statement

$\forall x.\ 0 < x \Rightarrow x2 \geq 1$ to be true.

However, the price is that some plausibly false statements turn out to be true; for example: $0 < 0 \Rightarrow 1 < 0$. Basically, in an absurd setting, everything is held to be true.

2. Sets

Aset is an collection of entities, oftenwrittenwith the syntax {e1, e2, . . . , en} when the set is finite. Making a set amounts to a decision to regard a collection of possibly disparate things as a single object. Here are some wellknown mathematical sets:

• B = {true, false}. The booleans, also known as the bit values. In situations where no confusion with numbers is possible, one could

have B = {0, 1}.

• N = {0, 1, 2, . . .}. The natural numbers.

• Z = {. . . ,−2,−1, 0, 1, 2, . . .}. The integers.

• Q = the rational (fractional) numbers.

• R = the real numbers.

• C = the complex numbers.

Note. Z, Q, R, and C will not be much used in the course, although Q and R will feature in one lecture.

There is a rich collection of operations on sets. Interestingly, all these operations are ultimately built from membership.

Membership of an element in a set The notation $a \in S$ means that a is a member, or element, of S. Similarly, a /∈ S means that a is not an element of S.

Equality of sets Equality of sets R and S is defined $R = S$ iff $(\forall x.\ x \in R$ iff $x \in S)$. Thus two sets are equal just when they have the same elements.

Note that sets have no intrinsic order. Thus {1, 2} = {2, 1}. Also,

sets have **no duplicates**. Thus {1, 2, 1, 1} = {2, 1}.

Subset R is a subset of S if every element of R is in S, but S may have extras. Formally, we write $R \subseteq S$ iff $(\forall x.\ x \in R \Rightarrow x \in S)$. Having $\subseteq$ available allows an (equivalent) reformulation of set equality: $R = S$ iff $R \subseteq S \wedge S \subseteq R$.

A few more useful facts about $\subseteq$:

• $S \subseteq S$, for every set S.

• $P \subseteq Q \wedge Q \subseteq R \Rightarrow P \subseteq R$

There is also a useful notion of proper subset: $R \subset S$ means that all elements of R are in S, but S has one or more extras. Formally,

R ⊂ S iff R ⊆ S ∧ R 6= S.

It is a common error to confuse $\in$ and $\subseteq$. For example, $x \in \{x, y, z\}$, but that doesn't allow one to conclude $x \subseteq \{x, y, z\}$. However, it is true that $\{x\} \subseteq \{x, y, z\}$

Union: The union of R and S, $R \cup S$, is the set of elements occurring in R

or S (or both). Formally, union is defined in terms of $\vee$: $x \in R \cup S$ iff $(x \in R \vee x \in S)$.

$\{1, 2\} \cup \{4, 3, 2\} = \{1, 2, 3, 4\}$

Intersection: The intersection of R and S, $R \cap S$, is the set of elements occurring in both R and S. Formally, intersection is defined in terms of $\wedge$:

$x \in R \cap S$ iff $(x \in R \wedge x \in S)$.

$\{1, 2\} \cap \{4, 3, 2\} = \{2\}$

Singleton sets A setwith one element is called a singleton. Note well that a singleton set is not the same as its element: $\forall$ x. x 6= {x}, even though $x \in \{x\}$, for any x.

Set difference $R - S$ is the set of elements that occur in R but not in S.Thus, $x \in R - S$ iff $x \in R \wedge x$ /∈ S. Note that S may have elements not in R. These are ignored. Thus

$\{1, 2, 3\} - \{2, 4\} = \{1, 3\}$.

Universe and complement Often we work in a setting where all sets are subsets of some fixed set U (sometimes called the universe). In that case we can write S to mean $U - S$. For example, if our universe is N, and Even is the set of even numbers, then Even is the set of odd numbers.

Example :. Let us take the Flintstone characters as our universe.

F = {Fred,Wilma, Pebbles,Dino} R = {Barney,Betty,BamBam}

U = F ∪ R ∪ {Mr . Slate}

Then we know

$\Phi = F \cap R$

because the two families are disjoint. Also, we can see that

F − {Fred,Mr . Slate} = {Wilma, Pebbles,Dino}.

What about Fred ⊆ F ? It makes no sense because Fred is not a set. The subset operation requires two sets. However, {Fred} ⊆ F is true; indeed

{Fred} ⊂ F.

We also know

{Mr . Slate, Fred} 6⊆ F

since Mr. Slate is not an element of F. Finally, we know that

F ∪ R = {Mr . Slate}

Remark. Set difference can be defined in terms of intersection and complement:

$A - B = A \cap B$

Empty set The symbol Φ stands for the empty set: the set with no elements. The notation {} may also be used. The empty set acts as an algebraic identity for several operations:

$\Phi \cup S = S$

$\Phi \cap S = \Phi$

$\Phi \subseteq S$

$\Phi - S = \Phi$

$S - \Phi = S$

$\overline{\Phi} = U$

Set comprehension This is also known as set builder notatation. The notation is

$$\{ \underbrace{\qquad}_{\text{template}} \mid \underbrace{\qquad}_{\text{condition}} \}$$

This denotes the set of all items matching the template, which also meet the condition. This, combined with logic, gives a natural way to concisely describe sets:

$\{x \mid x < 1\} = \{0\}$

$\{x \mid x > 1\} = \{2, 3, 4, 5, \ldots\}$

$\{x \mid x \in R \wedge x \in S\} = R \cap S$

$\{x \mid \exists y.x = 2y\} = \{0, 2, 4, 6, 8, \ldots\}$

$\{x \mid x \in U \wedge x \text{ is male}\} = \{\text{Fred}, \text{Barney}, \text{BamBam}, \text{Mr . Slate}\}$

Power set The set of all subsets of a set S is known as the powerset of S, written variously as P(S), Pow(S), or 2S.

$Pow(S) = \{s \mid s \subseteq S\}$

Example:

$Pow\{1, 2, 3\} = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$

If a finite set is of size n, the size of its powerset is 2n. A powerset is always larger than the set it is derived from, even if the orginal set is infinite.

Product of sets R×S, the product of two sets R and S, ismade by pairing each element of R with each element of S. Using set-builder notation, this can be concisely expressed:

$R \times S = \{(x, y) \mid x \in R \wedge y \in S\}$.

Example:

$$F \times R = \left\{ \begin{array}{l} (Fred, Barney),\ (Fred, Betty),\ (Fred, BamBam), \\ (Wilma, Barney),\ (Wilma, Betty),\ (Wilma, BamBam), \\ (Pebbles, Barney),\ (Pebbles, Betty),\ (Pebbles, BamBam), \\ (Dino, Barney),\ (Dino, Betty),\ (Dino, BamBam) \end{array} \right\}$$

In general, the size of the product of two sets will be the product of the sizes of the two sets.

Size of a set :The size of a set, also known as its cardinality, is just the number of elements in the set. It is common to write |A| to denote the cardinality of set A.

Example:

$|\ \{foo, bar, baz\}\ | = 3$

Cardinality for finite sets is straightforward; however it is worth noting that there is a definition of cardinality that applies to both finite and infinite sets: under that definition it can be proved that not all infinite sets have the same size! We will discuss this later in the course.

Summary of useful properties of sets:

Now we supply a few identities which are useful formanipulating expressions involving sets. The equalities can all be proved by expanding definitions. To begin with, we give a few simple facts about union, intersection, and the empty set.

$A \cup B = B \cup A$

$A \cap B = B \cap A$

$A \cup A = A$

$A \cap A = A$

$A \cup \Phi = A$

$A \cap \Phi = \Phi$

The following identities are associative, distributive, and absorptive properties:

$A \cup (B \cup C) = (A \cup B) \cup C$

$A \cap (B \cap C) = (A \cap B) \cap C$

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

$A \cup (A \cap B) = A$

$A \cap (A \cup B) = A$

The following identities are the so-called De Morgan laws, plus a few others.

$$\begin{aligned} \overline{A \cup B} &= \overline{A} \cap \overline{B} \\ \overline{A \cap B} &= \overline{A} \cup \overline{B} \end{aligned}$$

$$\begin{aligned} \overline{\overline{A}} &= A \\ A \cap \overline{A} &= \emptyset \end{aligned}$$

3. Functions

Informally, a function is a mechanism that takes an input and gives an output. One can also think of a function as a table, with the arguments down one column, and the results down another. In fact, if a function is finite, a table can be a good way to present it. Formally however, a function f is a set of ordered pairs with the property

$(a, b) \in f \wedge (a, c) \in f \Rightarrow b = c$

This just says that a function is, in a sense, univocal, or deterministic: there is only one possible output for an input. Of course, the notation $f(a) = b$ is preferred over $(a, b) \in f$. The domain and range of a function f are defined as follows:

Dom(f) = {x | ∃ y. f(x) = y}

Rng(f) = {y | ∃ x. f(x) = y}

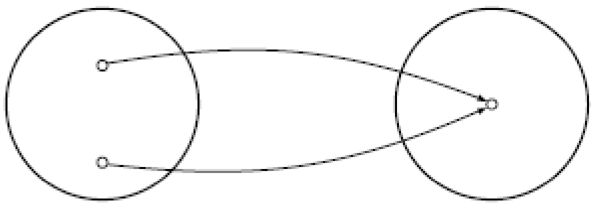
A common notation for specifying that function f has domain A and range B is the following:

f : A → B

Injective and Surjective functions: An injective, or one-to-one function sends different elements of the domain to different elements of the range:

$$\text{Injective}(f) \text{ iff } \forall x\ y.\ x \in \text{Dom}(f) \wedge y \in \text{Dom}(f) \wedge x \neq y \Rightarrow f(x) \neq f(y).$$

Pictorially, the following situation is avoided by an injective function:



An important consequence : if there's an injection from A to B, then B is at least the size of A. A surjective, or onto function is one in which every element of the range is produced by some application of the function:

Surjective(f : A → B) iff ∀ y. y ∈ B ⇒ ∃ x. x ∈ A ∧ f(x) = y

A bijection is a function that is both injective and surjective.

Closure :Closure is a powerful idea, and it is used repeatedly in this course. Suppose S is a set. If, for any x ∈ S, f(x) ∈ S, then S is said to be closed under f.

Example: N is closed under squaring:

∀ n. n ∈ N ⇒ n2 ∈ N.

2.3 Alphabets and Strings

An alphabet is a finite set of symbols, usually defined at the start of a problem statement. Commonly, ∑ is used to denote an alphabet.

Examples

∑ = {0, 1}

∑ = {a, b, c, d}

∑ = {foo, bar}

Non-examples

- N (or any infinite set)
- sets having symbolswith shared substructure, e.g., {foo, foobar}, since

this can lead to nasty, horrible ambiguity.

3. Strings

A string over an alphabet ∑ is a finite sequence of symbols from ∑. For example, if ∑ = {0, 1}, then 000 and 0100001 are strings over ∑. The strings provided in most programming languages are over the alphabet provided by the ASCII characters (and more extensive alphabets, such as Unicode, are common).

The empty string There is a unique string ε which is the empty string. There is an analogy between ε for strings and 0 for N. For example, both are very useful as identity elements.

Length :The length of a string s, written len(s), is obtained by counting each symbol from the alphabet in it. Thus, if ∑ = {f, o, b, a, r}, then

len(ε) = 0

len(foobar) = 6

but len(foobar) = 2, if ∑ = {foo, bar}.

Concatentation: The concatenation of two strings x and y just places them next to each other, giving the new string xy. If we needed to be precise, we could write x · y. Some properties of concatenation:

x(yz) = (xy)z associativity

xε = εx = x identity

len(xy) = len(x) + len(y)

The iterated concatenation x^n of a string x is the n-fold concatenation of x with itself.

Example : Let ∑ = {a, b}. Then

$(aab)^3$ = aabaabaab

$(aab)^1$ = aab

$(aab)^0$ = ε

The formal definition of x^n is by recursion:

$x^0 = ε$

$x^{n+1} = x^n \cdot x$

Notation. Repeated elements in a string can be superscripted, for convenience:

Example:

aab = a^2b.

Counting If x is a string over Σ and $a \in \Sigma$, then count(a, x) gives the number of occurrences of a in x:

count(0, 0010) = 3

count(1, 000) = 0

count(0, ε) = 0

The formal definition of count is by recursion:

count(a, ε) = 0

count(a, b · t) = if a = b then count(a, t) + 1 else count(a, t)

In the second clause of this definition, the expression (b · t) should be understood to mean that b is a symbol concatenated to string t.

Prefix: A string x is a prefix of string y iff there exists w such that y = x·w.

Eexample: **abaab** is a prefix of **abaab**aba. Some properties of prefix:

- ε is a prefix to every string
- x is a prefix to x, for any string x.
- A string x is a proper prefix of string y if x is a prefix of y and x ≠ y.

Reversal: The reversal xR of a string x = x1 ·. . . ·xn is the string xn ·. . . ·x1.

Pitfalls Here are some common mistakes people make when first confronted with sets and strings. All the following are true, but surprise some students.

• sets {a, b} = {b, a}

strings ab ≠ ba

• sets {a, a, b} = {a, b}

strings aab 6= ab

• $\underbrace{\emptyset}_{\text{empty set}} \neq \underbrace{\varepsilon}_{\text{empty string}} \neq \underbrace{\{\varepsilon\}}_{\text{singleton set holding empty string}}$

4. Languages

So much for strings. Now we discuss sets of strings, also called languages. Languages are one of the important themes of the course. We will start our discussion with Σ^*, the set of all strings over alphabet Σ. The set _∗ contains all strings that can be generated by iteratively

concatenating symbols from Σ, any number of times.

Example : If Σ = {a, b, c},

Σ^* = { ε , a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, aac, . . .}

If $\sum$ is a non-empty set, then $\sum^*$ is an infinite set, where each element is a finite string.

Set operations on languages

Now we apply set operations to languages; this will therefore be nothing new, since we've seen set operations already. However, it's worth the repetition.

Union

{a, b, ab} ∪ {a, c, ba} = {a, b, ab, c, ba}

Intersection

{a, b, ab} ∩ {a, c, ba} = {a}

Complement Usually, $\sum^*$ is the universe that a complement is takenwith respect to. Thus

$\bar{A}$ = {x ∈ | x /∈ A}

Example:

$\overline{\{x \mid len(x)\ is\ even\}}$ = {x ∈ $\sum^*$ | len(x) is odd}

Language reversal: The reversal of language A is written A^R and is defined (note the overloading)

$A^R = \{x^R \mid x \in A\}$

Language concatenation :The concatenation of languages A and B is defined:

AB = {xy | x ∈ A ∧ y ∈ B}

or using the 'dot' notation to emphasize that we are concatenating (note the overloading of ·):

A · B = {x · y | x ∈ A ∧ y ∈ B}

Example:

{a, ab} {b, ba} = {ab, abba, aba, abb}

Example: Two languages L1 and L2 such that L1 · L2 = L2 · L1 and L1 is not a subset of L2 and L2 is not a subset of L1 and neither language is {ε} are the following:

L1 = {aa} , L2 = {aaa}

Notes

- In general AB ≠ BA. Example: {a}{b} ≠ {b}{a}.
- A · ∅ = ∅ = ∅ · A.
- A · {ε} = A = {ε} · A.
- A · ε is nonsense—it's syntactically malformed.

Iterated language concatenation Well, if we can concatenate two languages, then we can certainly repeat this to concatenate any number of languages. Or concatenate a language with itself any number of times.

The operation An denotes the concatenation of A with itself n times. The formal definition is

$A^0 = \{\varepsilon\}$

$A^{n+1} = A \cdot A^n$

Another way to characterize this is that a string is in An if it can be split into n pieces, each of which is in A:

x ∈ An iff ∃w1 . . .wn. w1 ∈ A ∧ . . . ∧ wn ∈ A ∧ (x = w1 · · ·wn).

Example: Let A = {a, ab}. Thus $A^3 = A \cdot A \cdot A \cdot \{\varepsilon\}$, by unrolling the formal definition. To expand further:

A · A · A · {ε} = A · A · A

= A · {aa, aab, aba, abab} = {a, ab} · {aa, aab, aba, abab}

= {aaa, aaba, abaa, ababa, aaab, aabab, abaab, ababab}

Kleene's Star: It happens thatAn is sometimest limited because each string in it has been built by exactly n concatenations of strings from A. A more general operation, which addresses this shortcoming, is the so-called Kleene Star operation.1

$$\begin{aligned} A^* &= \textstyle\bigcup_{n\in\mathbb{N}} A^n \\ &= A^0 \cup A^1 \cup A^2 \cup \ldots \\ &= \{x \mid \exists n.\, x \in A^n\} \\ &= \{x \mid x \text{ is the concatenation of zero or more strings from } A\} \end{aligned}$$

Thus A* is the set of all strings derivable by any number of concatenations of strings in A. The notion of all strings obtainable by one or more concatenations of strings in A is often used, and is defined A+ = A · A∗,

$$A^+ = \bigcup_{n>0} A^n = A^1 \cup A^2 \cup A^3 \cup \ldots$$

An = A1 ∪ A2 ∪ A3 ∪ . . .

Example:

A = {a, ab} A* = A0 ∪ A1 ∪ A2 ∪ . . .

= {ε} ∪ {a, ab} ∪ {aa, aab, aba, abab} ∪ . . .

A+ = {a, ab} ∪ {aa, aab, aba, abab} ∪ . . .

Some facts about Kleene star:

- The previously introduced definition of _∗ is an instance of Kleene star.
- ε is in A∗, for every language A, including Ø∗ = {ε}.
- L ⊆ L∗.

Example: An infinite language L over {a, b} for which L 6= L^* is the following:

$L = \{a^n \mid n \text{ is odd}\}$.

A common situation when doing proofs about Kleene star is reasoning with formulas of the form $x \in L*$, where L is a perhaps complicated expression.

A useful approach is to replace $x \in L*$ by $\exists n.x \in Ln$ before proceeding.

Example : Prove $A \subseteq B \Rightarrow A* \subseteq B*$.

Proof. Assume $A \subseteq B$. Now suppose that $w \in A^*$. Therefore, there is an n such that $w \in A^n$. That means $w = x1 \cdot \ldots \cdot xn$ where each $xi \in A$. By the assumption, each $xi \in B$, so $w \in B^n$, so $w \in B^*$.

That concludes the presentation of the basic mathematical objects we will be dealing with: strings and their operations; languages and their operations Summary of useful properties of languages Since languages are just sets of strings, there are a few others:

$A \cdot (B \cup C) = (A \cdot B) \cup (A \cdot C)$

$(B \cup C) \cdot A = (B \cdot A) \cup (C \cdot A)$

$A \cdot (B0 \cup B1 \cup B2 \cup \ldots) = (A \cdot B0) \cup (A \cdot B1) \cup (A \cdot B2) \ldots$

$(B0 \cup B1 \cup B2 \cup \ldots) \cdot A = (B0 \cdot A) \cup (B1 \cdot A) \cup (B2 \cdot A) \cup \ldots$

$A^{**} = (A^*)^* = A^*$

$A^* \cdot A^* = A^*$

$A^* = \{\varepsilon\} \cup A^+$

$\emptyset^* = \{\varepsilon\}$