CheatSheet AICC I

$\mathbb{BANGBANG}$, yyp from Joachim Favre

2022 – 2023

1 Logical Equivalence

Equivalences with basic connectives

Equivalence	Name
$p \wedge T \equiv p$ $p \vee F \equiv p$	Identity laws
$p \vee T \equiv T$ $p \wedge F \equiv F$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p)$	Double negation law
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv T$ $p \wedge \neg p \equiv F$	Negation laws
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws
$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's Law
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws

Equivalences with implications

$$p \implies q \equiv \neg p \lor q$$
$$p \implies q \equiv \neg q \implies \neg p$$

$$p \lor q \equiv \neg p \implies q$$
$$p \land q \equiv \neg\,(p \implies \neg q)$$
$$\neg\,(p \implies q) \equiv p \land \neg q$$

$$(p \implies q) \land (p \implies r) \equiv p \implies (q \land r)$$
$$(p \implies r) \land (q \implies r) \equiv (p \lor q) \implies r$$
$$(p \implies q) \lor (p \implies r) \equiv p \implies (q \lor r)$$
$$(p \implies r) \lor (q \implies r) \equiv (p \land q) \implies r$$

$$p \iff q \equiv (p \implies q) \land (q \implies p)$$
$$p \iff q \equiv \neg p \leftrightarrow \neg q$$
$$p \iff q \equiv (p \land q) \lor (\neg p \land \neg q)$$
$$\neg\,(p \iff q) \equiv p \leftrightarrow \neg q$$

1.1 CNF and DNF

Construction of DNF

1. Construct the truth table for the proposition.
2. Select the rows that evaluate to T.
3. For each of the propositional variables in the selected rows, add a conjunction which includes the positive form of the propositional if the variable is assigned T in that row, or the negated form if the variable is assigned F in that row.

Construction of CNF

1. Find the DNF for the proposition.
2. Use De Morgan's Law to move Negations inside ().
3. Use distributive and associative laws to form the CNF.

2 Predicate Logic

The only important formulas :

$$\begin{aligned} \neg\exists! x P(x) &\equiv \forall x(\neg P(x) \lor \exists y(P(y) \land y \neq x)) \\ \forall x(P(x) \land Q(x)) &\equiv \forall x P(x) \land \forall x Q(x) \\ \exists x(P(x) \land Q(x)) &\not\equiv \exists x P(x) \land \exists x Q(x) \\ \forall x(P(x) \lor Q(x)) &\not\equiv \forall x P(x) \lor \forall x Q(x) \\ \forall x(P(x) \implies Q(x)) &\not\equiv \forall x P(x) \implies \forall x Q(x) \end{aligned}$$

3 Proofs

Deduction	Corresponding tautology	Name
p q $\therefore\ p \land q$	$p \land q \implies p \land q$	Conjunction
$p \implies q$ p $\therefore\ q$	$(p \land p \implies q) \implies q$	Modus Ponens
$p \implies q$ p $\therefore\ q$	$(\neg q \land p \implies q) \implies \neg p$	Modus Tollens
$p \implies q$ $q \implies r$ $\therefore\ p \implies r$	$(p \implies q \land q \implies r) \implies (p \implies r)$	Hypothetical Syllogism
$p \implies q$ $q \implies r$ $\therefore\ p \implies r$	$((\neg p \lor r) \land (p \lor q)) \implies (q \lor r)$	Resolution
$p \lor q$ $\neg p$ $\therefore\ q$	$((p \lor q) \land \neg p) \implies q$	Disjunctive Syllogism
p $\therefore\ p \lor q$	$p \implies (p \lor q)$	Addition
$p \land q$ $\therefore\ p$	$(p \land q) \implies p$	Simplification

4 Relations

4.1 Binary Relation Definition

Let A and B be sets. A binary relation $\mathcal{R}$, from A to B is a subset of $A \times B$.

4.2 Reflexive Relations

A relation $\mathcal{R}$ on a set A is reflexive iff

$$\forall a(a \in A \implies (a,a) \in \mathcal{R})$$

Examples

$\mathcal{R}_1 = \{(1,1),(1,2),(2,1),(3,4),(4,4)\}$ is not reflexive because $(2,2) \notin \mathcal{R}$
$\mathcal{R}_2 = \{(1,1),(2,1),(1,2),(2,2)\}$ is reflexive because both $(1,1)$ and $(2,2)$ are in $\mathcal{R}$
$\mathcal{R}_3 = \{(a,b)|a\ divides\ b\}$ is reflexive because $\forall a, a|a$

4.3 Symmetric Relations

A relation $\mathcal{R}$ on a set A is symmetric iff

$$\forall a \forall b((a,b) \in \mathcal{R} \implies (b,a) \in \mathcal{R})$$

Examples

$\mathcal{R}_1 = \{(1,1),(1,2),(2,1),(3,4),(4,4)\}$ is not symmetric because $(4,3) \notin \mathcal{R}$
$\mathcal{R}_2 = \{(1,1),(2,1),(1,2),(2,2)\}$ is symmetric
$\mathcal{R}_3 = \{(a,b)|a\,divides\,b\}$ is not symmetric because $4|2$ but $2 \not| 4$

4.4 AntiSymmetric Relations

A relation $\mathcal{R}$ on a set A is antisymmetric iff

$$\forall a \forall b(((a,b) \in \mathcal{R} \wedge (b,a) \in \mathcal{R}) \implies a = b)$$

Examples

$\mathcal{R}_1 = \{(2,1),(2,2)\}$ is antisymmetric
$\mathcal{R}_2 = \{(2,1),(1,2),(2,2)\}$ is not antisymmetric
$\mathcal{R}_3 = \{(a,b)|a = b+1\}$ is antisymmetric because the LHS of the implication is false !

4.5 Transitive Relation

A relation $\mathcal{R}$ on a set A is transitive, iff

$$\forall a \forall b \forall c (((a,b) \in R \wedge (b,c) \in \mathcal{R}) \implies (a,c) \in \mathcal{R})$$

Examples

$$\mathcal{R} = \{(a,b) | a \leq b\} \quad \text{is transitive.}$$

4.6 Equivalence Relations

A relation on a set A is called equivalence relation if it is reflexive, symmetric, and transitive.

Two elements a and b that are related by an equivalence relation are called equivalent. We use the notation $a \sim b$

Examples

$$\mathcal{R} = \{(a,b) \in \mathbb{R} \times \mathbb{R} | a - b \in \mathbb{Z}\}$$

$$\begin{aligned}
\text{reflexive} &: a - a = 0 \in \mathbb{Z} \\
\text{symmetric} &: a - b \in \mathbb{Z} \implies b - a \in \mathbb{Z} \\
\text{transitive} &: a - b \in \mathbb{Z} \wedge b - c \in \mathbb{Z} \implies (a-b) + (b-c) = a - c \in \mathbb{Z}
\end{aligned}$$

4.7 Equivalence Classes

Let $\mathcal{R}$ be an equivalence relation on a set A. The set of all elements that are related to an element $a \in A$ is called the equivalence class of a noted $[a]_{\mathcal{R}}$.

$$[a]_{\mathcal{R}} = \{b | (a,b) \in \mathcal{R}\}$$

Theorem If $\mathcal{R}$ is an equivalence relation on a set A then :

$$\mathcal{R}(a,b) \iff [a]_R = [b]_R \iff [a]_R \cap [b]_R \neq \emptyset$$

$\mathcal{R}(a,b)$ means that a and b are related. ($\mathcal{R}(a,b)$ is the same as $\mathcal{R}(b,a)$, if $\mathcal{R}$ is an equivalence relation.)

Theorem about partition of a set Let $\mathcal{R}$ be an equivalence relation on a set S. The the equivalence classes of $\mathcal{R}$ form a partition of S.

Examples The 3 congruence classes $[0]_3, [1]_3, [2]_3$ form a partition of $\mathbb{Z}$.

$$[0]_3 = \{..., -6, -3, 0, 3, 6, ...\}$$
$$[1]_3 = \{..., -5, -2, 1, 4, 7, ...\}$$
$$[2]_3 = \{..., -4, -1, 2, 5, 8, ...\}$$

4.8 Partial Ordering and poset

A relation $\preceq$ on a set S is called a partial ordering or partial order if it is reflexive, antisymmetric, and transitive. A set S together with a partial ordering $\preceq$ is called a partially ordered set (a.k.a. poset) and is denoted $(S, \preceq)$

Examples $(\mathbb{Z}, \geq)$ is a poset, $(\mathbb{Z}^+, |)$ is a poset, $(\mathcal{P}(S), \subseteq)$ is a poset.

4.9 Total Ordered and Well-ordered sets

If $(S, \preceq)$ is a poset and every pair of elements are comparable then it is totally ordered.
If every subset of a totally-ordered poset $(S, \preceq)$ has a least element then it is well-ordered.

Examples $(\mathbb{N}, \leq)$ is well-ordered, $(\mathbb{R}^+, \geq)$ is totally-ordered, $(\mathbb{Z}^+, |)$ is not totally-ordered.

4.10 Lattices

A poset in which every pair of elements has a least upper bound **and** a greatest lower bound is called a lattice.

Examples $(\mathbb{Z}^+, |)$ is a lattice where the greatest lower bound is $\gcd(a, b)$ and the least upper bound is $lcm(a, b)$.

$(\mathcal{P}(S), \subseteq)$ is a lattice where the greatest lower bound is $A \cap B$ and the least upper bound is $A \cup B$.

5 Countability

A countable set S is either finite or has the same cardinality as $\mathbb{Z}$, that is, there exists a bijection $\mathbb{Z} \to S$. The cardinality of an infinite set that is countable is $|S| = |\mathbb{Z}| = \aleph_0$.

The set of real numbers $\mathbb{R}$ is uncountable. Soo to prove that a set A is uncountable we can show that there is an injection from $\mathbb{R} \to A$.

6 Sequences

Find a recurrnce relation from closed formula and the other way around

1. Try to find a commun difference (one that is close enough).
2. Try to find a commun ratio (one that is close enough).
3. Look for an intuitive relation, add variables and solve for them.
4. Look for well known relations.

Sum	**Closed From**
$\sum_{k=0}^{n} ar^k \quad (r \neq 0)$	$\frac{ar^{n+1} - a}{r - 1}$
$\sum_{k=1}^{n} k$	$\frac{n(n+1)}{2}$
$\sum_{k=1}^{n} k^2$	$\frac{n(n+1)(2n+1)}{6}$
$\sum_{k=1}^{n} k^3$	$\left(\frac{n(n+1)}{2}\right)^2$
$\sum_{k=0}^{\infty} x^k \quad \lvert x\rvert < 1$	$\frac{1}{1-x}$
$\sum_{k=1}^{\infty} kx^{k-1} \quad \lvert x\rvert < 1$	$\frac{1}{(1-x)^2}$

7 Big-O, Big-Ω, Big-Θ, little-o

Let f and g be funtions from the set of integers or the set of real numbers to the set of real numbers. We say that $f(x)$ is $O(g(x))$ if $\exists C$ and $\exists k$ such that:

$$|f(x)| \leq C|g(x)|, \quad \forall x > k$$

$$\begin{aligned} f(x) \text{ is } \Omega(g(x)) &\iff g(x) \text{ is } O(f(x)). \\ f(x) \text{ is } \Theta(g(x)) &\iff f(x) \text{ is } O(g(x)) \text{ and } \Omega(g(x)). \\ f(x) \text{ is } o(g(x)) &\iff \lim_{x\to\infty} \frac{f(x)}{g(x)} = 0 \end{aligned}$$

commun functions ($c > 1$)

$$1 \prec \log\log n \prec \log^c n \prec n^{1/c} \prec n \log n \prec n^c \prec c^n \prec n! \prec n^n$$

8 Algorithms

8.1 Binary Search

Algorithm 1: Binary Search

Data: x, $a_1 < a_2 < \cdots < a_n$

$lower\_bound \leftarrow 1$;
$upper\_bound \leftarrow n$;
while $lower\_bound < upper\_bound$ **do**
    $m \leftarrow \lfloor \frac{lower\_bound+upper\_bound}{2} \rfloor$;
    if $x > a_m$ **then**
        $lower\_bound \leftarrow m + 1$;
    else
        $upper\_bound \leftarrow m$;
    end
end
if $x = a_{lower\_bound}$ **then**
    $index \leftarrow lower\_bound$;
else
    $index \leftarrow 0$;
end

Result: index (index of x if $x = a_i$ else 0)

The algorithmic complexity of Binary Search is $\Theta(\log(n))$.

8.2 Bubble Sort

Algorithm 2: Bubble Sort

Data: $a_1, a_2, \ldots, a_n$

for $i \leftarrow 1$ **to** $n - 1$ **do**
    for $j \leftarrow 1$ **to** $n - i$ **do**
        if $a_j > a_{j+1}$ **then**
            swap a_j and a_{j+1};
        end
    end
end

Result: $b_1 < b_2 < \cdots < b_n$ (the list a_i sorted)

Bubble Sort preforms $\frac{n(n-1)}{2}$ comparisons, $\Theta(n^2)$ swaps and the algorithmic complexity is $\Theta(n^2)$.

8.3 Selection Sort

Algorithm 3: Selection Sort

Data: $a_1, a_2, \ldots, a_n$

for $i \leftarrow 1$ **to** $n - 1$ **do**
    min $\leftarrow i + 1$;
    for $j \leftarrow i + 1$ **to** n **do**
        if $a_{min} > a_j$ **then**
            min $\leftarrow j$;
        end
    end
    if $a_i > a_{min}$ **then**
        swap a_i and a_{min};
    end
end

Result: $b_1 < b_2 < \cdots < b_n$ (the list a_i sorted)

Selection Sort performs $\frac{n(n-1)}{2}$ comparisons, $\Theta(n)$ swaps and the algorithmic complexity is $\Theta(n^2)$.

8.4 Insertion Sort

Algorithm 4: Insertion Sort

Data: $a_1, a_2, \ldots, a_n$

for $j \leftarrow 2$ **to** n **do**
    $i \leftarrow 1$;
    while $a_j > a_i$ **do**
        $i \leftarrow i + 1$;
    end
    $m \leftarrow a_j$;
    for $k \leftarrow 0$ **to** $j - i - 1$ **do**
        $a_{j-k} \leftarrow a_{j-k-1}$;
    end
    $a_i \leftarrow m$;
end

Result: $b_1 < b_2 < \cdots < b_n$ (the list a_i sorted)

Insertion Sort preforms $\Theta(n^2)$ comparisons, $\Theta(n^2)$ swaps and the algorithmic complexity is $\Theta(n^2)$.

8.5 Stable matching

Algorithm 5: Gale-Shapley

Data: A_1, A_2 such that $|A_1| = |A_2|$, $A = A_1 \cup A_2$
L_n the preferences of each $n \in A_i$

$M \leftarrow \emptyset$;
while $|M| < |A_1|$ **do**
  $x \leftarrow$ an unpaired element of A_1;
  $y \leftarrow$ the first element of L_x: $L_x[1] \in A_2$;
  if $\exists(x', y)$ *i.e. y is paired with x'* **then**
    if *y prefers x' i.e. x' comes before x in L_y* **then**
      remove $L_x[1]$;
    else
      remove $L_{x'}[1]$;
      replace (x', y) by (x, y) in M;
    end
  else
    add (x, y) in M;
  end
end
Result: M

Out of all possible matchings, Gale-Shapley's algorithm gives a maximum stable matching which is X-optimal when the elements of X "propose". The algorithmic complexity is $\Theta(|A|^2)$.