SPENGERGASSE
ausbildung mit zukunft

POS1
logic

by Mag. Harald Zumpf

propositional logic

- logic: use of valid reasoning
- in propositional logic sentences can be true or false.
  (even if we don't know the exact answer)

examples for sentences

- Vienna is the capital of Austria: **true**
- Graz is in the north of Vienna: **false**
- move on!: command -> not allocatable
- do you play golf?: question -> not allocatable

binary logic

- a sentence is either true **or** false
- true or false are called **logical values or truth assignments**

notation (how you write it down)

- propositional variables are shown as small letters (**p, q, r, s**)
- **boolean terms**, sentences, etc. are shown as small letters (**a, b, c, d**)
- **logical values**: positive (**t, +, 1**,...); negative (**f, -, 0**,...)

boolean operations

- sentences, boolean terms and propositional variables can be linked together by junctions (linking devices): **and**, **or**, **if then**, ...

truth table

a	$\neg a$
t	f
f	t

Negation (inversion) of a

negation (inversion) $\neg$

- negation of a sentence
- example: he did go to school.
  negation: he did **not** go to school.

notation (how you write it down)

- he did win the game. term a
- he did **not** win the game. **negation** of term a

$$\neg\, a$$

boolean operation (negation) → $\neg$

term → a

conjunction (and) $\wedge$

- boolean operation (junction) for linking two sentences (terms)
- example: this drink consists of wine **and** this drink consists of water.
- true results only if **both** terms are true

a	b	$a \wedge b$
t	t	t
t	f	f
f	t	f
f	f	f

a **and** b

disjunction (or) $\vee$

- boolean operation (junction) for linking two sentences (terms)
- example: its allowed to sing **or** dance.
- true results if at least one term is true

a	b	$a \vee b$
t	t	t
t	f	t
f	t	t
f	f	f

a **or** b

exclusive or (either or) $\bar{\vee}$

- boolean operation (junction) for linking two sentences (terms)
- example: **Either** i go out, **or** i stay at home.
- true results only if one term is true

a	b	$a \bar{\vee} b$
t	t	f
t	f	t
f	t	t
f	f	f

either a **or** b

(material) implication (if then) ->

- boolean operation (junction) for linking two sentences (terms)
- example: **if** it rains, **then** i will stay at home.
  (if it does not rain you can stay at home too)
- true results: if a is true the value of b is taken
  as a result. If a is false the result is always true.

a	b	a -> b
t	t	t
t	f	f
f	t	t
f	f	t

conclusion

premise

read as:
if a **then** b,
a **implies** b

equivalence or boolean equality (equals) <->

- boolean operation (junction) for linking two sentences (terms)
- example: his behavior **equals** hers.
- true results only if a and b are **equal**

a	b	a <-> b
t	t	t
t	f	f
f	t	f
f	f	t

a **equals** b

non-equivalence (unequal) <≠>

- boolean operation (junction) for linking two sentences (terms)
- example: he is **unequal** to his twin brother.
- true results only if a and b are **unequal**

a	b	a <≠> b
t	t	f
t	f	t
f	t	t
f	f	f

a **unequal** b

boolean operators - software engineering

junction	Pascal or Fortran	logical operator (C)	binary operator (C)
negation	NOT	!	~
conjunction	AND	&&	&
disjunction	OR	||	|
non-equivalence	XOR	not existing	^

propositional formulas

- links any number of sentences (terms) with boolean operators
- tasks with any complexity can be constructed with propositional formulas

how to create a boolean formula (term)

- analyze the sentences and define them with small letter (p, q, r, s) in alphabetical order
- define truth values (t, f) in the **truth table** for all possible combinations (permutate)
- number of possible combinations (lines): $2^{\text{amount of sentences}}$

1. analyze and split sentences

example

- sentence: **if** it rains, **then** the road **or** the roof is wet.
- **p** it rains, ...
- **q** ...the road is wet.
- **r** ...the roof is wet.

2. define the formula

- p **->** (q **v** r)

if p **then** q **or** r

3. creating the truth table

example

- one column for each boolean variable
  rows: $2^{\text{amount of sentences}}$
- one column for the created formula
- one column for each partial result
- all possible truth value combinations of truth values (p, q, ...)
- calculate partial results (from inside to the outside)
- define final result

3. creating the truth table

example

p	q	r	p -> (q v r)	
t	t	t	t	t
t	t	f	t	t
t	f	t	t	t
t	f	f	f	f
f	t	t	t	t
f	t	f	t	t
f	f	t	t	t
f	f	f	t	f

homework

example

- create truth tables for:

  If I practice, I get good marks and I am satisfied.

  Susi only visits the party, if Hans visits the party and Peter does not visit the party.

  Only if Peter and Hans wear red jackets then they are twins, or they have the same taste.

special terms

name	property	example
tautology	term is always true	$p \lor \neg p$
contradiction	term is always false	$p \land \neg p$
indeterminism	term can be true or false	$p \rightarrow (q \lor r)$

tautologies and indeterminisms are called **compliable terms**

boolean substitution

- a boolean variable is substitutable by an other boolean term.
- substitutions of tautological parts by other tautologies create tautologies
- substitutions of contradictional parts by other contradictions create contradictions

notation (how you write it down)

- **a [p/b]** **every** variable p in term a is substituted by term b

boolean substitution

example

- a: p -> q -> p
- b: (r v s)
- a[p/b]: (r v s) -> q -> (r v s)

$$\text{a[p/b]: } \overset{\text{r v s}}{\overset{\downarrow}{\text{p}}} \text{ -> q -> } \overset{\text{r v s}}{\overset{\downarrow}{\text{p}}}$$

partial formulas (sub-formulas)

- every sub-formula is a sub-formula of itself
- every sub-formula of a sub-formula is a sub-formula of the given formula
- if two formulas are equal each sub-formula can be substituted by a sub-formula of the other formula
- like in math: equal things can be substituted by equal things.

boolean replacement

example

- Equal things can be substituted by equal things.
- a <-> (¬c -> ¬b)
  b <-> (a ∧ c)
  c <-> ¬ a
- replace c in row 2 by ¬ a
- new row 2: b <-> (a ∧ ¬ a)
- (a ∧ ¬ a) is a contradiction, result: f
- new row 2 results in a contradiction
- new row 1: a <-> (¬ f -> ¬ f), result: **a is true**

important equalities

example

a <-> b; b <-> a	reflexive identity	a ∧ (b ∧ c) <-> (a ∧ b) ∧ c	associativity of conjunctions
¬ ¬ a <-> a	law of double negation	a ∧ (b v c) <-> (a ∧ b) v (a ∧ c)	distributivity of conjunctions
a v a <-> a	idempotence of disjunctions	a ∧ (a v b) <-> a	law of absorption
a v t <-> t	idempotence of disjunctions	¬ (a ∧ b) <-> (¬ a v ¬ b)	law of DeMorgan
a v f <-> a	idempotence of disjunctions	a -> b <-> ¬ a v b	law of DeMorgan
a v b <-> b v a	commutativity of disjunctions	a -> (b -> c) <-> b -> (a -> c)	commutative law for premises
a v (b v c) <-> (a v b) v c	associativity of disjunctions	a <-> b <-> b <-> a	associativity identity
a v (b ∧ c) <-> (a v b) ∧ (a v c)	distributivity of disjunctions	a ∧ b <-> ¬ (a -> ¬ b)	derivation with DeMorgan
a v (a ∧ b) <-> a	law of absorption	a ∧ t <-> a	idempotence of conjunction
¬ (a v b) <-> (¬ a ∧ ¬ b)	law of DeMorgan	a ∧ f <-> f	idempotence of conjunction
a ∧ a <-> a	idempotence of conjunctions	a ∧ b <-> b ∧ a	commutativity of conjunction
a ↓ a <-> ¬ a	idempotence of Peirce Operators	a ↑ a <-> ¬ a	idempotence of exclusion

<->, v, ∧ are associative and commutative junctions

Peirce-arrow NOR (neither) $\downarrow$

- boolean operation (junction) for linking two sentences (terms) by **neither**
- example: My pants are **neither** black **or** white
- true results if both variables (terms) are false

a	b	$a \downarrow b$
t	t	f
t	f	f
f	t	f
f	f	t

a **NOR** b,
a **neither** b

exclusion NAND - Sheffer stroke (not and) $\uparrow$

- boolean operation (junction) for linking two sentences (terms) by **not both** or **not and**
- example: **Not** Susi **and** Jenny should come to my party.
  example: Not both girls (Susi and Jenny) should come to my party.
- false results if **both** variables (terms) are true.

a	b	a ↑ b
t	t	f
t	f	t
f	t	t
f	f	t

not (a **and** b**)**
a **nand** b

conversion of NAND ↑ example

- term: p -> q should be created with the exclusive use of NAND

  p -> q <-> ¬(p ∧ ¬q) important equivalences

  p -> q <-> ¬(p ∧ (q ↑ q)) idempotence of exclusion

  p -> q <-> p ↑ (q ↑ q)

- alternative:

  p -> q <-> ¬p v q important equivalences

  p -> q <-> ¬p v ¬¬q double negation

  p -> q <-> ¬(p ∧ ¬q) law of DeMorgan

  p -> q <-> ¬(p ∧ (¬(q ∧ q))) q ∧ q <-> q

  p -> q <-> p ↑ (q ↑ q)

solving common problems with propositional logic

example

- A detective has to solve the following case:
- If R is guilty then P is guilty too.
- If Q or R are guilty, P is not guilty.
- If P or R are not guilty, then Q is guilty.
- -> Solve the case and prove it with the truth table.

solving common problems with propositional logic **example**

- A detective has to solve the following case:
- If R is guilty then P is guilty too. $r \rightarrow p$
- If Q or R are guilty, P is not guilty. $(q \lor r) \rightarrow \neg p$
- If P or R are not guilty, then Q is guilty. $(\neg p \lor \neg r) \rightarrow q$
- -> Solve the case and prove it with the truth table.

$$(r \rightarrow p) \land ((q \lor r) \rightarrow \neg p) \land ((\neg p \lor \neg r) \rightarrow q)$$

solving common problems with propositional logic **example**

(r -> p) ∧ (q v r) -> ¬p ∧ (¬p v ¬r) -> q **Q is guilty!**

p	q	r	(r -> p)	∧	(q v r)	->	¬p	∧	(¬p v ¬r)	-> q
t	t	t	t	f	t	f	f	f	f	t
t	t	f	t	f	t	f	f	f	t	t
t	f	t	t	f	t	f	f	f	f	t
t	f	f	t	t	f	t	f	f	t	f
f	t	t	f	f	t	t	t	f	t	t
f	t	f	t	t	t	t	t	t	t	t
f	f	t	f	f	t	t	t	f	t	f
f	f	f	t	t	f	t	t	f	t	f
			1.	**7.**	**2.**	**4.**	**3.**	**8.**	**5.**	**6.**

solving common problems with propositional logic example

- Find out who is a liar / tells the truth:
- Anton says: If Christian is a liar, then Berta is a liar.
- Berta says: Anton and Christian tell the truth.
- Christian says: Anton is a liar.
- -> Use the truth table.

solving common problems with propositional logic **example**

- Tautology, Indeterminism or Contradiction?

$A \rightarrow B \leftrightarrow \neg B \wedge A$

$((A \rightarrow B) \wedge \neg B) \rightarrow \neg A$

$A \rightarrow B \leftrightarrow \neg B \rightarrow \neg A$

$A \wedge \neg(\neg A \rightarrow B)$

$((A \rightarrow B) \wedge (B \rightarrow C)) \rightarrow (A \rightarrow C)$

$((A \rightarrow B) \wedge (B \rightarrow C)) \leftrightarrow \neg(A \rightarrow B) \vee \neg(B \rightarrow C)$

solving common problems with propositional logic example

- **Does Doris have either a mouse or a hamster?**
- Astrid says: If Doris has a hamster, not both of the others tell the truth.
- Bettina says: Doris tells the truth and Clemens lies.
- Clemens says: The other three tell the truth.
- Doris says that she owns a mouse.

solving common problems with propositional logic example

- Only if it does not get cold, it will neither be windy nor rainy.
- In general there will only be wind when it gets cold.
- But if there is no wind, it will surely rain and be cold.
- And if it stays dry (no rain), there will be wind or it will not get cold.

How will the weather be?

solving common problems with propositional logic example

- The fire department is alerted to an operation. When the emergency vehicle arrives at the scene, the firefighters already see that there is a fire on the second floor. But before they can take further steps, the group commander has to get an approximate picture and therefore asks a few witnesses and learns that a 3 room apartment is burning. He still questions the inhabitants of the apartment, who say the following:
- Residents 1: When I went out I only saw how the living room and the dining room were burning.
- Residents 2: I am sure that if the dining room is on fire, the rest of the rooms are also burning.
- Residents 3: Only if the bedroom is not burning, then the dining room is burning but not the living room.
- Which rooms are burning?

solving common problems with propositional logic example

- If Philipp wears a blue suit, then he either puts on a tie or a bow tie.

- Philipp only wears a gilet, if he wears a blue suit and either a tie or a bow tie.
- Philipp only wears a tie, if he wears a blue suit or a blue suit and a gilet.
- Only if Philipp wears neither a blue suit or a gilet, he wears a bow tie.
- If Philipp wears a bow tie, then he either wears a gilet or a tie.

What will Philipp wear this evening?

normal forms

- normal forms are methods of unification for logical terms and sentences
- a **literal** is a boolean variable (negated or not negated)
- a **minterm** conjuncts up to an unlimited number of literals
  example: $p \land \neg q \land t \land \neg q \land p$ literals can be equal
- a **maxterm** disjuncts up to an unlimited number of literals
  example: $p \lor \neg q \lor t \lor \neg q \lor p$ literals can be equal
- **a disjunctive normal form (DNF)** is a disjunction of minterms
  example: $(p \land \neg q) \lor (q \land t) \lor (\neg q \land p) \lor \ldots$
- **a conjunctive normal form (CNF)** is a conjunction of maxterms
  example: $(p \lor \neg q) \land (q \lor t) \land (\neg q \lor p) \land \ldots$

SPENGERGASSE
ausbildung mit zukunft

normal forms

- a **base conjunction** is a minterm, where boolean variables are unique (and there is no t or f)
  example: $p \wedge \neg q \wedge r$

- a **base disjunction** is a maxterm, where boolean variables are unique (and there is no t or f)
  example: $p \vee \neg q \vee r$

- a **canonical** DNF (**CDNF**) is a disjunction of base conjunctions, where every base conjunction is unique
  example: $(p \wedge \neg q \wedge r) \vee (\neg p \wedge q \wedge r) \vee (p \wedge q \wedge \neg r)$

- a **canonical** CNF (**CCNF**) is a conjunction of base disjunctions, where every base disjunction is unique
  example: $(p \vee \neg q \vee r) \wedge (\neg p \vee q \vee r) \wedge (p \vee q \vee \neg r)$

by Mag. Harald Zumpf

creating a CDNF

example

- 1. create a truth table with the given formula
  example: (¬p -> q) v r

p	q	r	(¬p -> q)	(¬p -> q) v r
t	t	t	t	t
t	t	f	t	t
t	f	t	t	t
t	f	f	t	f
f	t	t	t	t
f	t	f	t	t
f	f	t	f	t
f	f	f	f	f

creating a CDNF

example

- mark and number every row where results are true

p	q	r	(¬p -> q)	(¬p -> q) v r	
t	t	t	t	t	**1.**
t	t	f	t	t	**2.**
t	f	t	t	t	**3.**
t	f	f	t	t	**4.**
f	t	t	t	t	**5.**
f	t	f	t	t	**6.**
f	f	t	f	t	**7.**
f	f	f	f	f	

creating a CDNF

example

3. create a base conjunction out of all literals (p, q, r) of the marked and numbered rows. Use the given literal when its true, negate it if it's false

p	q	r	(¬p -> q) v r			
t	t	t	t	t	**1.**	p ∧ q ∧ r v
t	t	f	t	t	**2.**	p ∧ q ∧ ¬r v
t	f	t	t	t	**3.**	p ∧ ¬q ∧ r v
t	f	f	t	t	**4.**	p ∧ ¬q ∧ ¬r v
f	t	t	t	t	**5.**	¬p ∧ q ∧ r v
f	t	f	t	t	**6.**	¬p ∧ q ∧ ¬r v
f	f	t	f	t	**7.**	¬p ∧ ¬q ∧ r
f	f	f	f	f		

p ∧ q ∧ r v p ∧ q ∧ ¬r vp ∧ ¬q ∧ r vp ∧ ¬q ∧ ¬r v¬p ∧ q ∧ r v¬p ∧ q ∧ ¬r v¬p ∧ ¬q ∧ r

by Mag. Harald Zumpf

creating a CCNF

example

- 1. create a truth table with the given formula
  example: (¬p -> q) v r

p	q	r	(¬p -> q)	(¬p -> q) v r
t	t	t	t	t
t	t	f	t	t
t	f	t	t	t
t	f	f	t	t
f	t	t	t	t
f	t	f	t	t
f	f	t	f	t
f	f	f	f	f

by Mag. Harald Zumpf

creating a CCNF

example

- mark and number every line which results false

p	q	r	$(\neg p \rightarrow q)$	$(\neg p \rightarrow q) \lor r$	
t	t	t	t	t	
t	t	f	t	t	
t	f	t	t	t	
t	f	f	t	t	
f	t	t	t	t	
f	t	f	t	t	
f	f	t	f	t	
f	f	f	f	f	**1.**

creating a CCNF

example

- 3. create a base disjunction out of all literals (p, q, r) of the marked and numbered rows. Use the given literal when its false, negate it if it's true

p	q	r	(¬p -> q) v r		
t	t	t	t	t	
t	t	f	t	t	
t	f	t	t	t	
t	f	f	t	t	
f	t	t	t	t	
f	t	f	t	t	
f	f	t	f	t	
f	f	f	f	f	**1.** p v q v r

p v q v r

KV-diagram

- the KV-diagram visualizes logical terms and can be used for simplification and optimization
- it gives us a shortened and optimized logical term
- it converts a DNF (disjunctive normal form) into a visual model
- there are 2^n segments for n variables
- neighboring segments differ in only one literal
- the neighboring segments of boundary segments are the opposing segments

visualization of a KV-diagram

- 2 variables:

	p	
q	$p \land q$	$\neg p \land q$
	$p \land \neg q$	$\neg p \land \neg q$

visualization of a KV-diagram

- 3 variables:

p

q	$p \land q \land \neg r$	$p \land q \land r$	$\neg p \land q \land r$	$\neg p \land q \land \neg r$
	$p \land \neg q \land \neg r$	$p \land \neg q \land r$	$\neg p \land \neg q \land r$	$\neg p \land \neg q \land \neg r$

r

visualization of a KV-diagram

- 4 variables:

p

q

$p \land q \land \neg r \land \neg s$	$p \land q \land r \land \neg s$	$\neg p \land q \land r \land \neg s$	$\neg p \land q \land \neg r \land \neg s$
$p \land q \land \neg r \land s$	$p \land q \land r \land s$	$\neg p \land q \land r \land s$	$\neg p \land q \land \neg r \land s$
$p \land \neg q \land \neg r \land s$	$p \land \neg q \land r \land s$	$\neg p \land \neg q \land r \land s$	$\neg p \land \neg q \land \neg r \land s$
$p \land \neg q \land \neg r \land \neg s$	$p \land \neg q \land r \land \neg s$	$\neg p \land \neg q \land r \land \neg s$	$\neg p \land \neg q \land \neg r \land \neg s$

s

r

by Mag. Harald Zumpf

creation of a KV-diagram

example

given truth table:

p	q	r	s	result
t	t	t	t	f
t	t	t	f	f
t	t	f	t	f
t	t	f	f	f
t	f	t	t	f
t	f	t	f	f
t	f	f	t	t
t	f	f	f	t
f	t	t	t	t
f	t	t	f	t
f	t	f	t	t
f	t	f	f	t
f	f	t	t	t
f	f	t	f	t
f	f	f	t	t
f	f	f	f	t

creation of a KV-diagram

example

- put the truth values into the KV-diagram

p	q	r	s	result
t	t	t	t	f
t	t	t	f	f
t	t	f	t	f
t	t	f	f	f
t	f	t	t	f
t	f	t	f	f
t	f	f	t	t
t	f	f	f	t
f	t	t	t	t
f	t	t	f	t
f	t	f	t	t
f	t	f	f	t
f	f	t	t	t
f	f	t	f	t
f	f	f	t	t
f	f	f	f	t

p

q

s

f	f	t	t
f	f	t	t
t	f	t	t
t	f	t	t

r

by Mag. Harald Zumpf

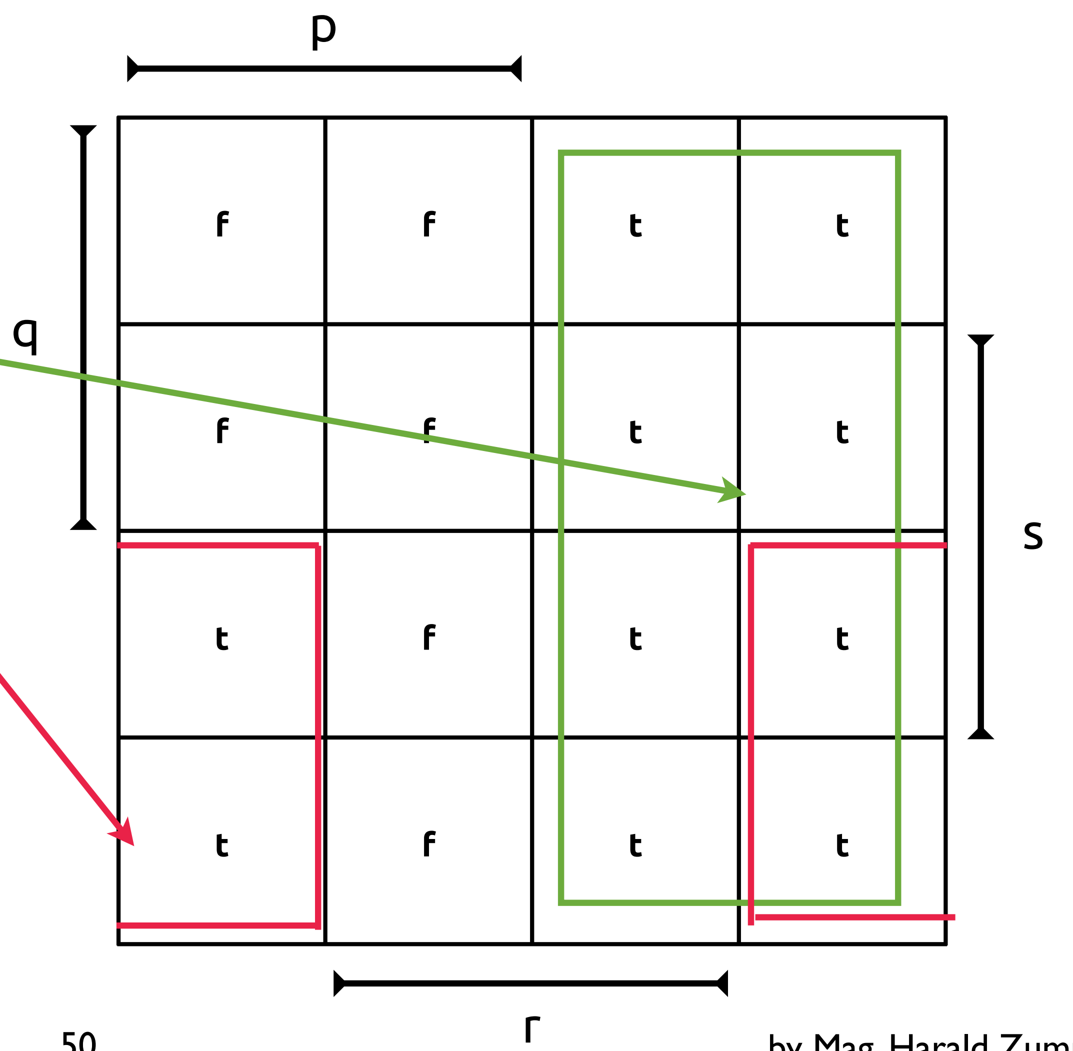
creation of a KV-diagram

example

- find (t)-segments - as big as possible and link them with a disjunction:

1. segment: ¬p

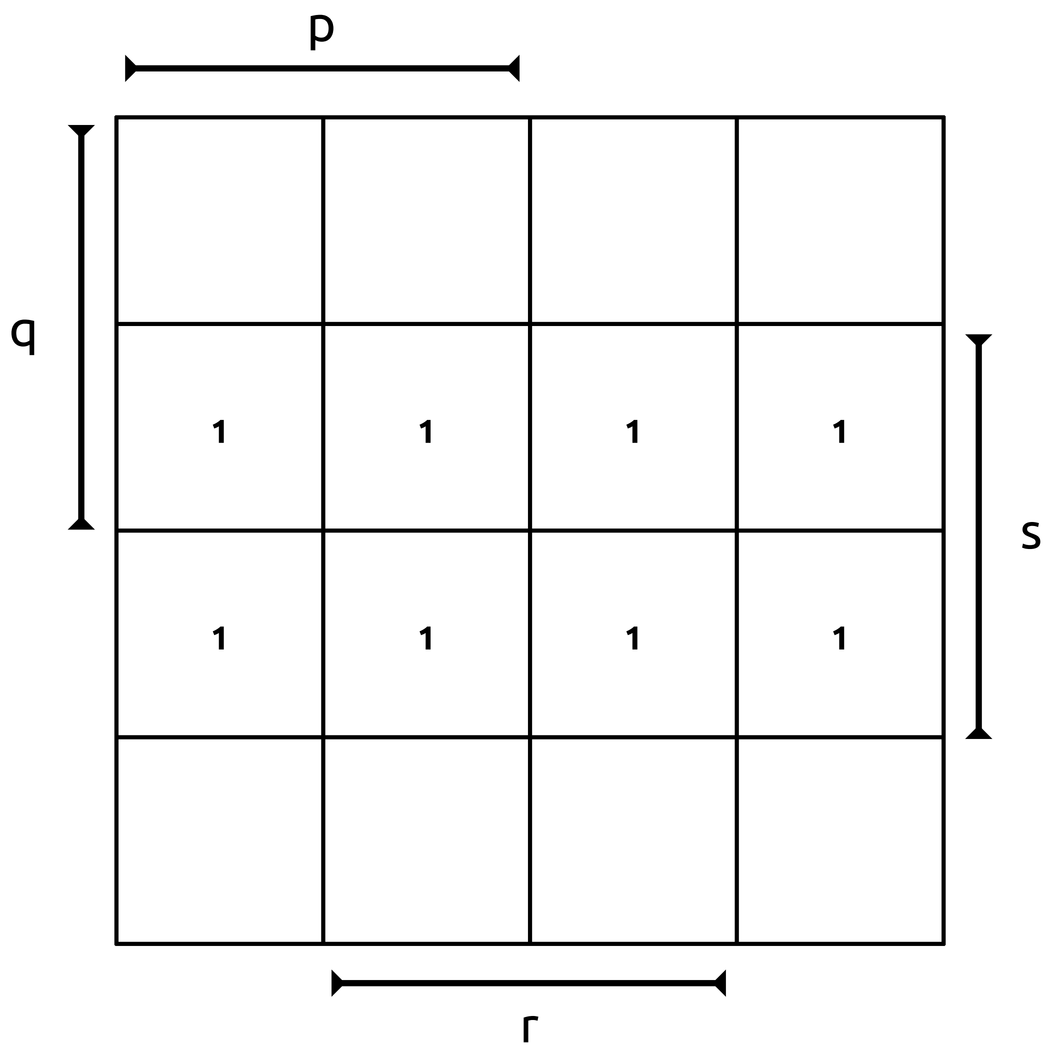
2. segment: ¬q ∧ ¬r



result: ¬p v (¬q ∧ ¬r)
with usage of
DeMorgan: ¬(p ∧ (q v r))

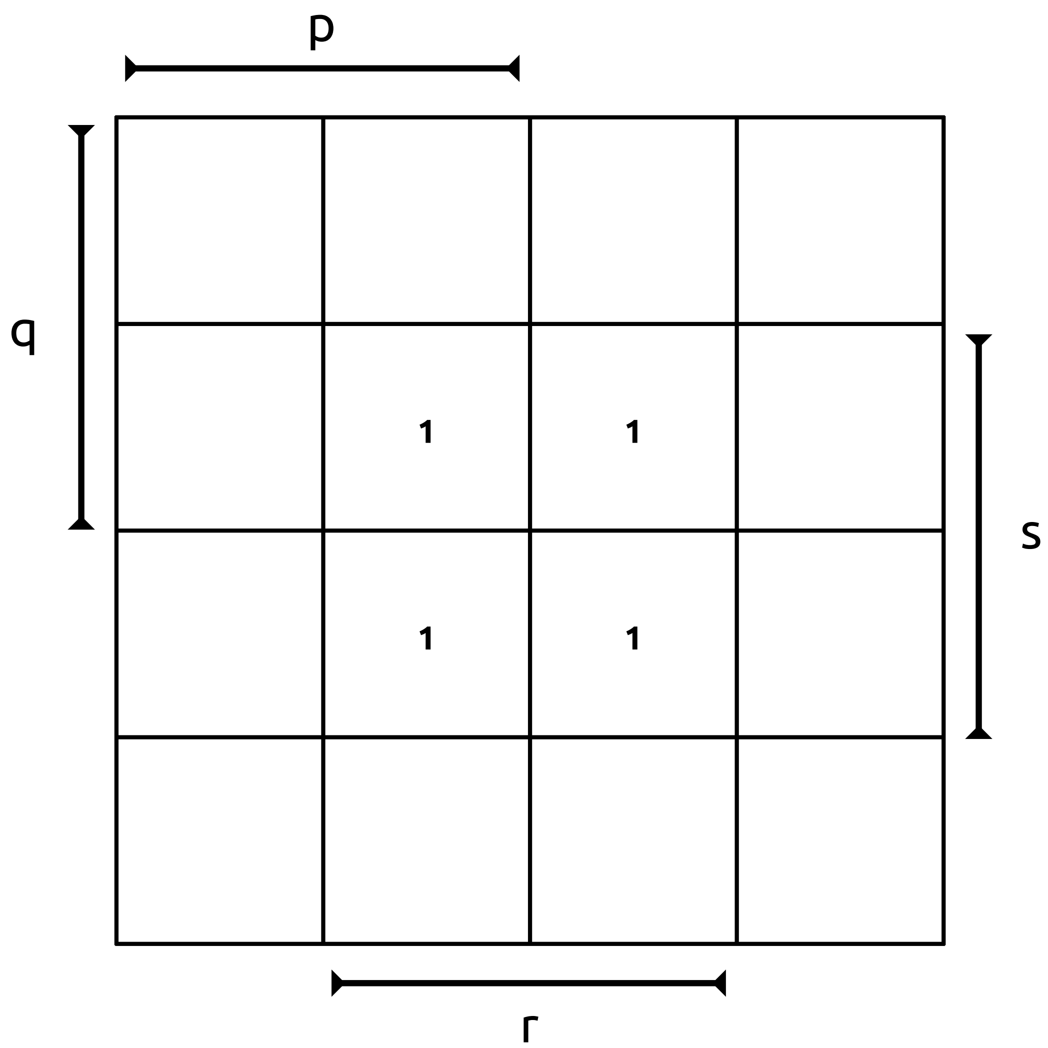
law of reduction

- result: **s**



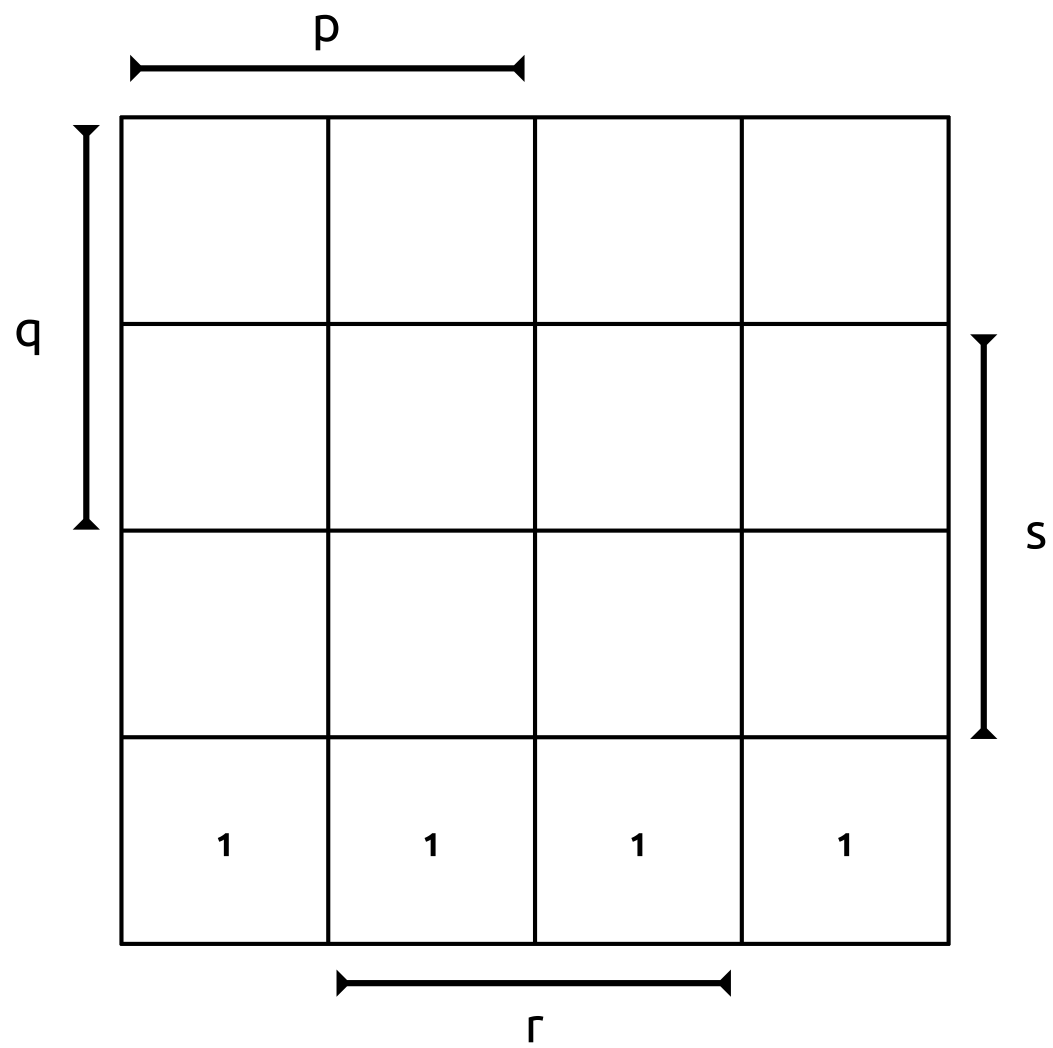
law of reduction

- result: **r** ∧ **s**



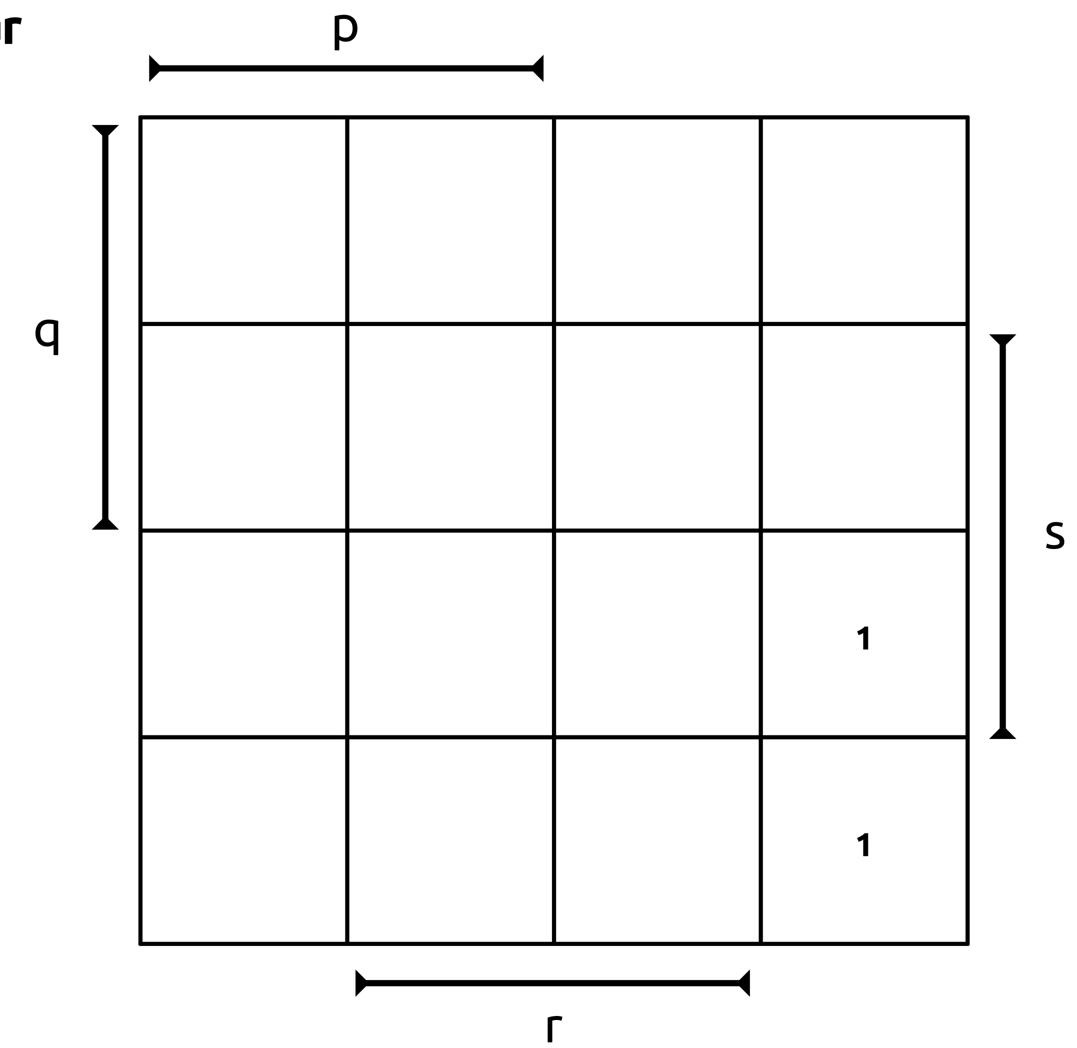
law of reduction

- result: **¬q** ∧ **¬s**



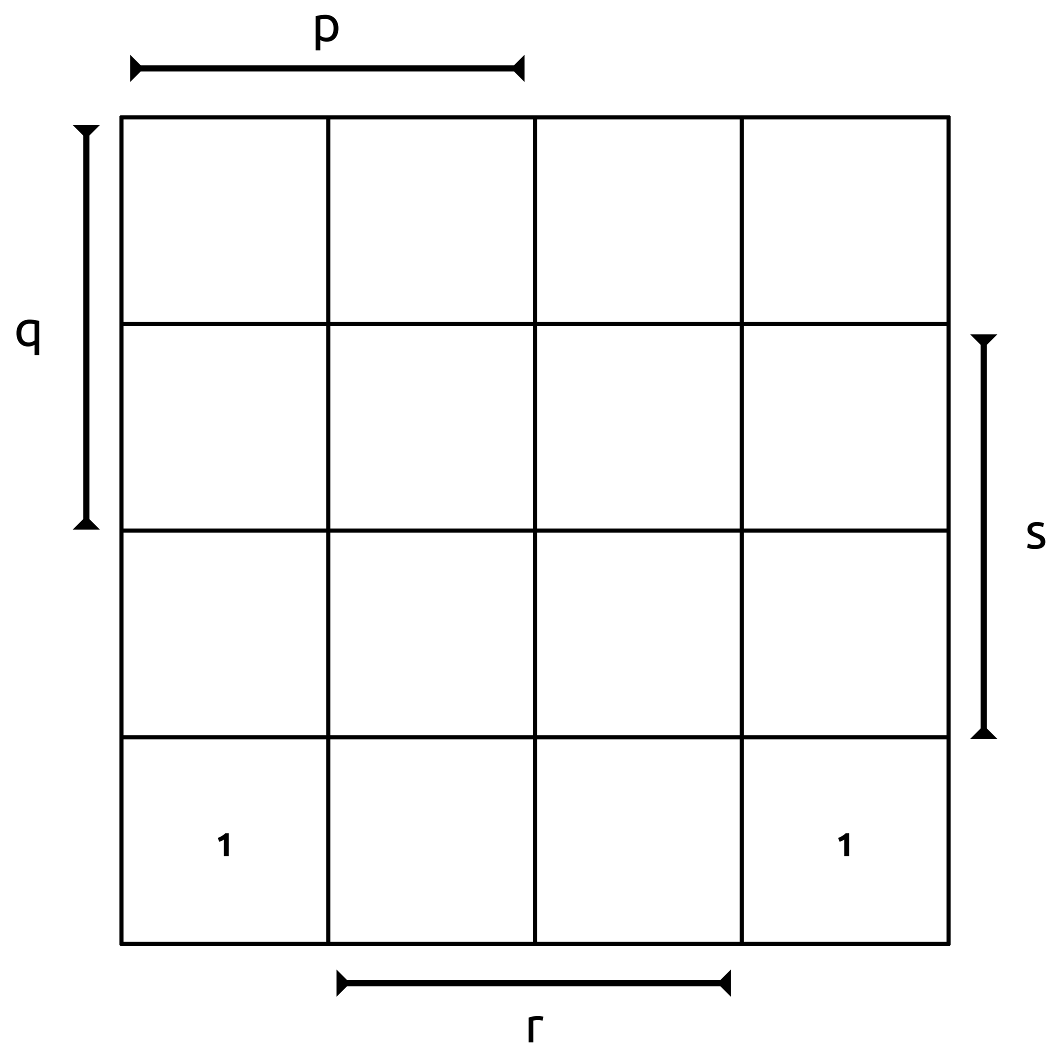
law of reduction

- result: **¬p** ∧ **¬q** ∧ **¬r**



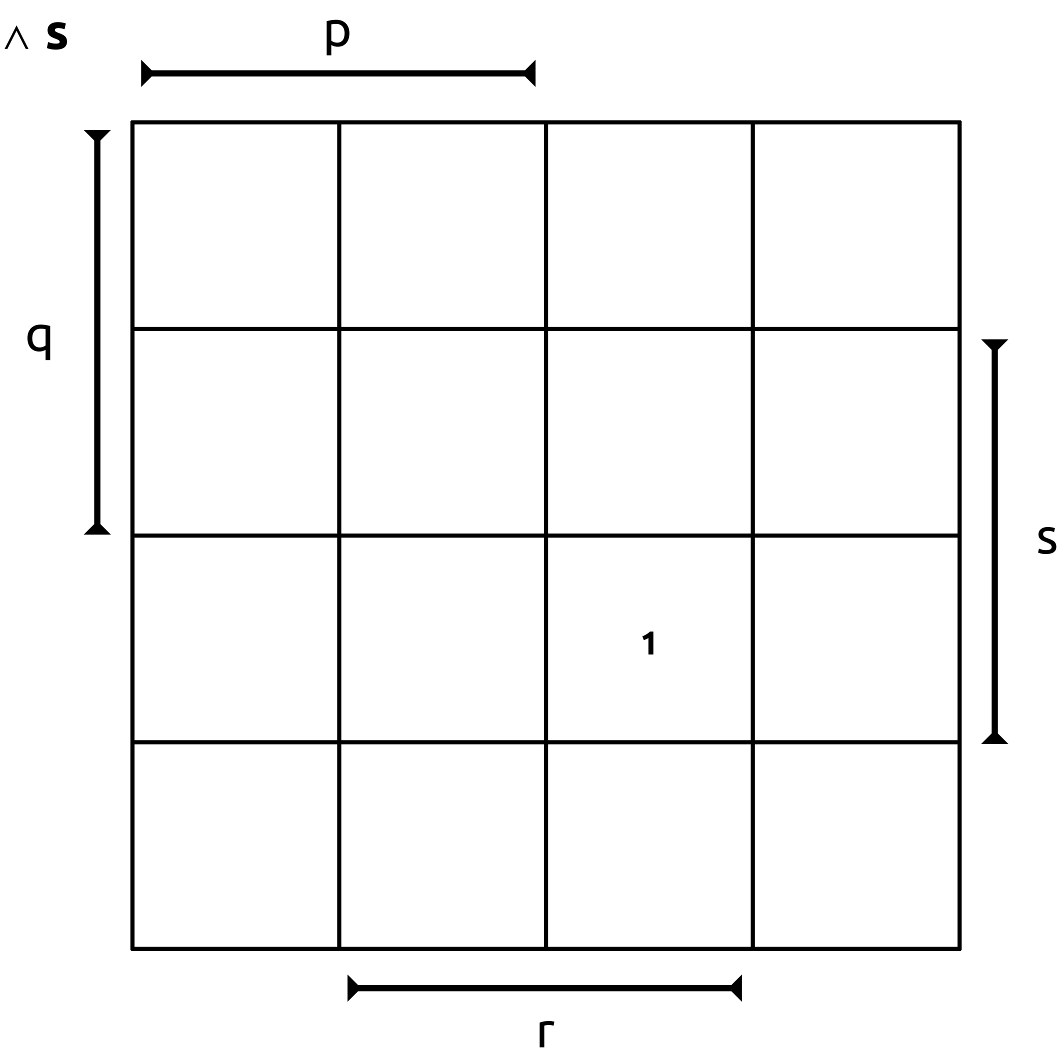
law of reduction

- result: **¬r** ∧ **¬s** ∧ **¬q**



law of reduction

- result: $\neg \mathbf{p} \wedge \neg \mathbf{q} \wedge \mathbf{r} \wedge \mathbf{s}$



practical implication with the KV-diagram

example

- sink the ship - grid:

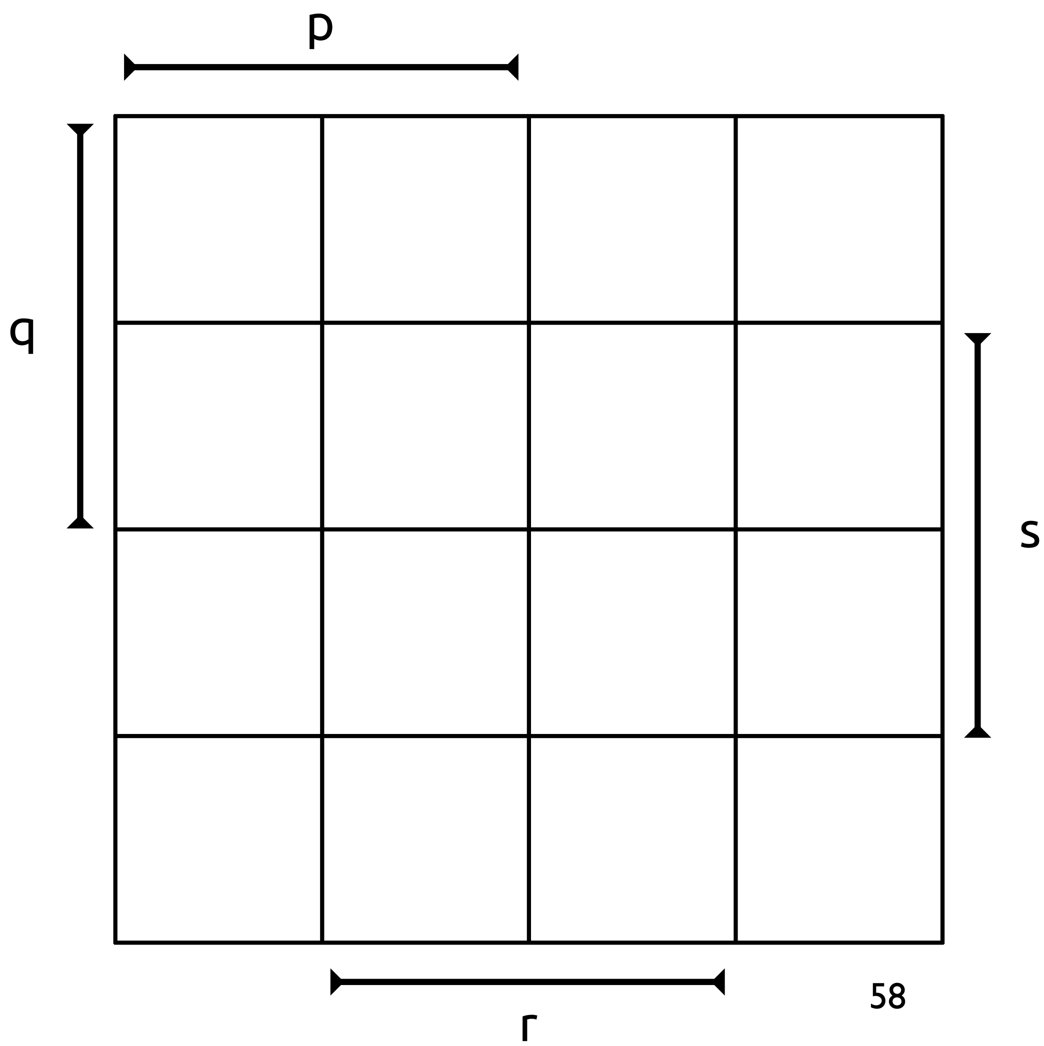
4				
3				
2				
1				
	A	B	C	D

memory usage
for allocation (ASCI):
8 x 1 Byte = **8 Byte**

practical implication with the KV-diagram

example

- sink the ship - grid with KV-diagram:

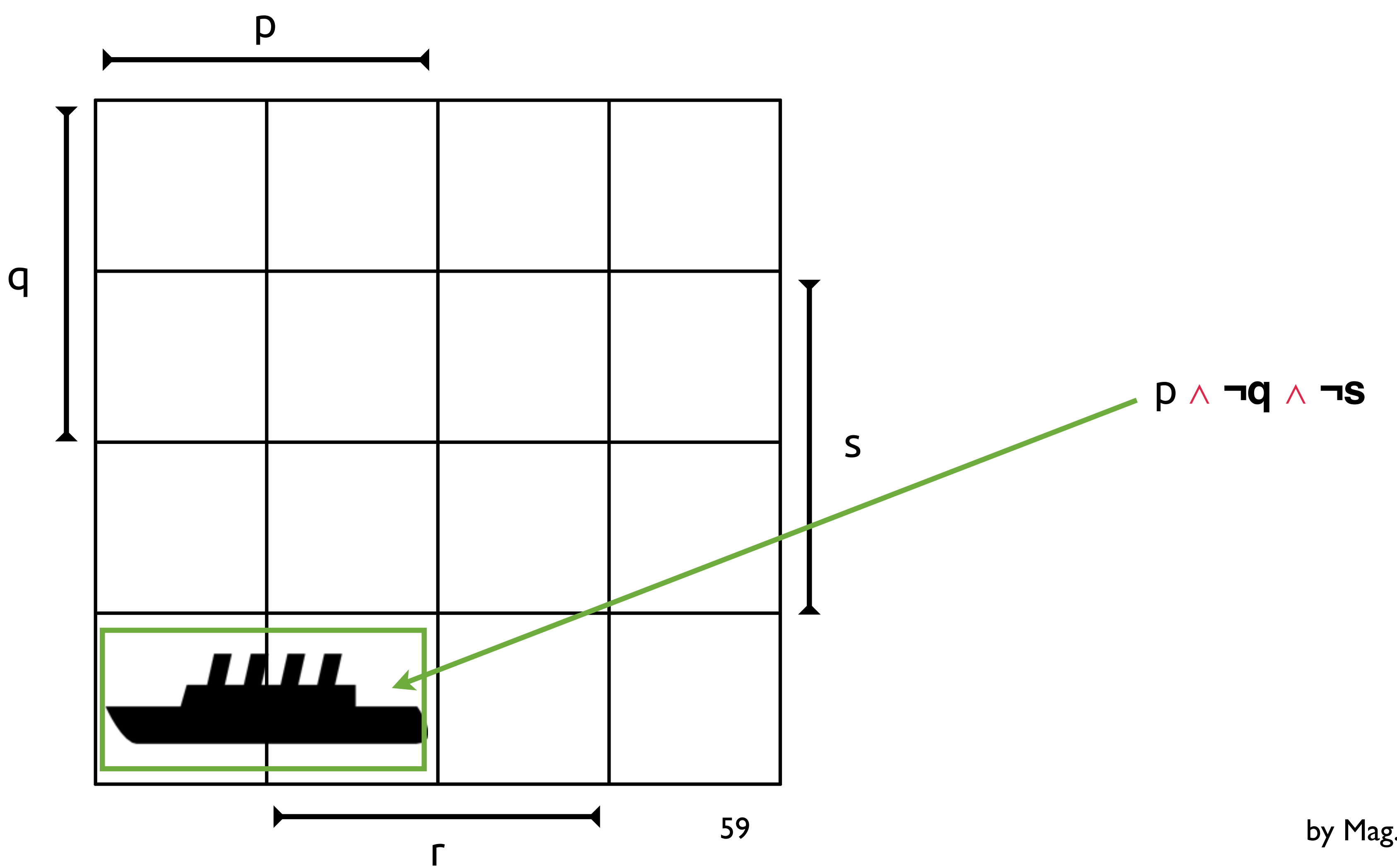


memory usage
for allocation (ASCI):
4 x 1 Byte = **4 Byte**

practical implication with the KV-diagram

example

- sink the ship - grid with KV-diagram:



system of addition

- roman numbering system

I	V	X	L	C	D	M
1	5	10	50	100	500	1000

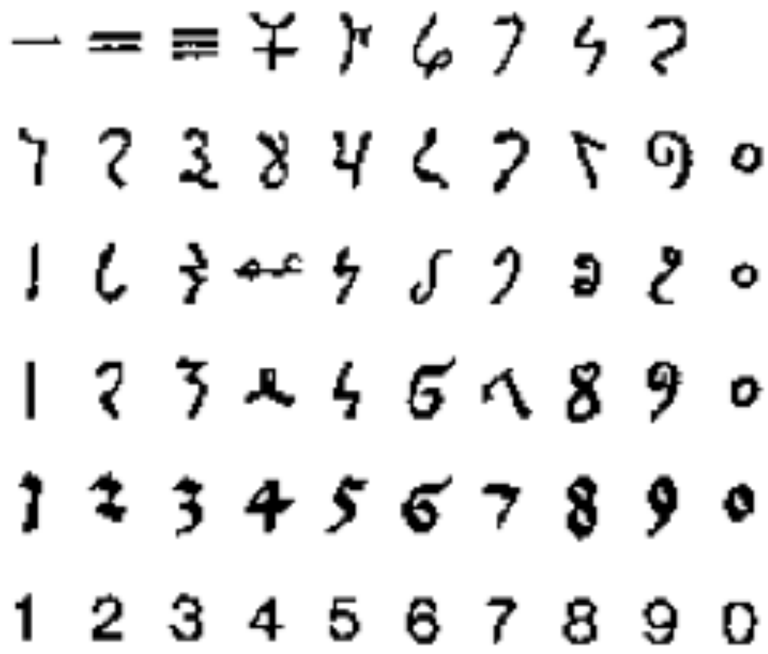
- example: MDCLXVI = 1666

denominational number system (our system)

- usage of digits (0, 1, 2, ... ,9)
- decimal system
- combination of digits through groups
- the position of a digit defines its place value
- the unit position is represented by the most right digit
- 2348 = 2*thousand + 3*hundred + 4*ten +8*one

denominational number system (our system)

- historical development



India 300 BC

India 800 AD

Western Arabia 1100 AD

Europe 1500 AD

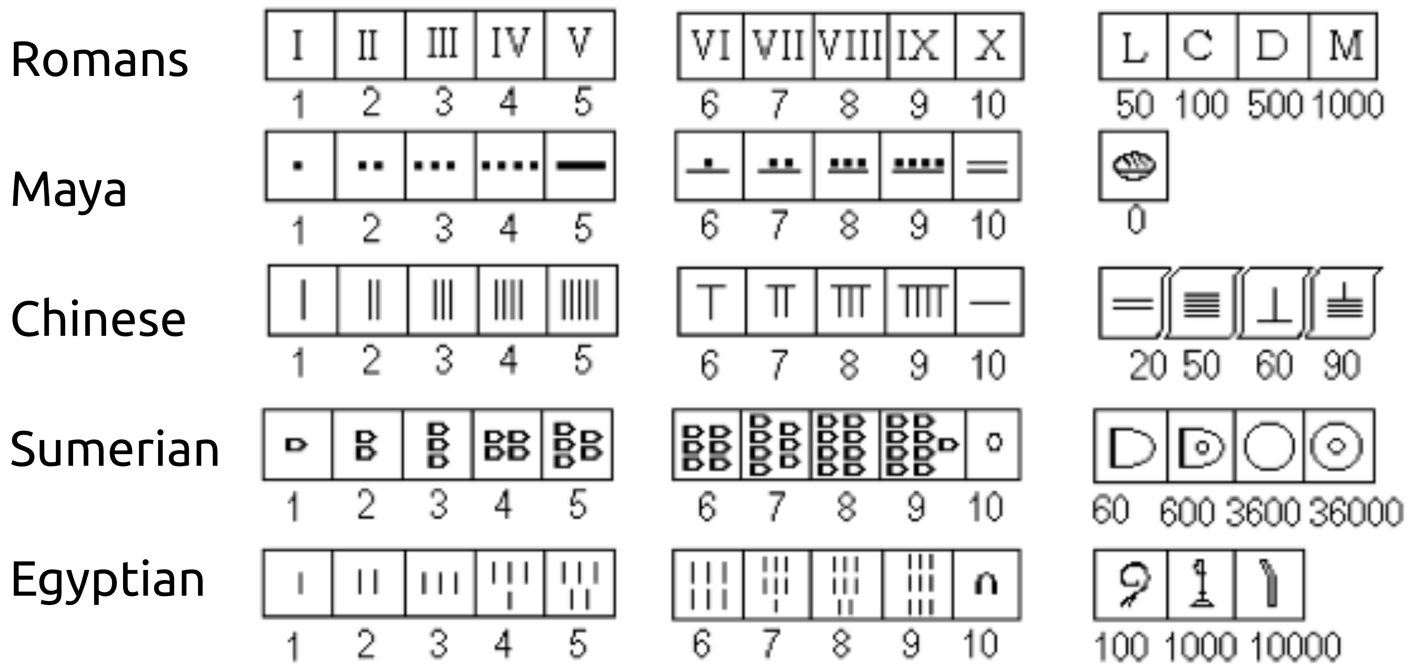
Europe 1600 AD

Today

from: www.rechenhilfsmittel.de

denominational number system (our system)

- digits from different cultures



from: www.rechenhilfsmittel.de

denominational number system (our system)

- from which culture?

from: www.wikipedia.com

numeral systems with various basis

- decimal system: basis 10
- binary system: basis 2
- octal system: basis 8
- hexadecimal system: basis 16
- the lower the basis the longer the number

variation of digits

- the variation of digits is represented by the different symbols which can be used to write down a number
- the number of different digits equals the basis
- if the basis B > 10 the variation of digits is extended by letters
  B = 16: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, **A, B, C, D, E, F**}

SPENGERGASSE
ausbildung mit zukunft

representation of values in various systems

binary system	octal system	decimal system	hexadecimal system
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F
10000	20	16	10

numeral systems with basis B

- every value can be represented in every numeral system
- the representation of a value with digits in a denominational system can only be done with B > 1
- 2345 = 2*(10^3) + 3*(10^2) + 4*(10^1) + 5*(10^0)
  note: 10^0 = 1 !
- 0,778 = 0*(10^0) + 7*(10^-1) + 7 *(10^-2) + 8*(10^-3)
- $$Z = \sum_{i=-m}^{n} z_i \cdot B^i$$

notation

ordinal number of the most significant place
$0 \leq n < \infty$

ordinal number of the least significant place
$1 \leq m < \infty$

$$Z = \sum_{i=-m}^{n} z_i \cdot B^i$$

value **digit** **basis** **index**

notation

example

- Z = **931,5706**
- B = 10
- n = 2 (note: start counting with 0) $0 \leq n < \infty$
- m = 4 (note: start counting with 1) $1 \leq m < \infty$
- integral part: $Z_{gz} = 931$
- fractional part: $Z_{gb} = 5706$
- $z_2=9$; $z_1=3$; $z_0=1$; $z_{-1}=5$; $z_{-2}=7$; $z_{-3}=0$; $z_{-4}=6$
- $Z_{gz} = \sum_{i=0}^{2} z_i \cdot 10^i$ $\quad Z_{gb} = \sum_{i=-4}^{-1} z_i \cdot 10^i$ $\quad Z = Z_{gz} + Z_{gb}$

exercise / homework

example

- **define Z, B, m, n, Z_{gz} , Z_{gb} of:**
- $456,36_8$
- $10010,11_2$
- $AB79,AE1_{16}$

the modulo division (tool) %

- the modulo division is a mathematical operation similar to the common division
- the modulo division of two values (normal numbers) delivers the remainder of the division as a result (normal number)
- we use the symbol % for the operator
- example:

8 % 3 = 2 8 **modulo** 3

addition

example

basis 10:

		7	7	6
+		5	7	8
	1	3	5	4

- 6 + 8 = 14, write 4 down, carry over 1
  when the result of the addition is larger than basis B (10):
  write down: z_i (4), z_{i+1} (1) is carried over
- 7 + 7 = 14 + 1 (carry over) = 15, 5 and 1 carried over
- 7 + 5 = 12 + 1 (carry over) = 13, 5 and 1 carried over
- 0 + 0 = 0 +1 (carry over) = 1
- if there are no more digits to add, the operation is over

by Mag. Harald Zumpf

addition

example

basis 8:

		1	7	4
+		6	1	4
	1	0	1	0

- 4 + 4 = 10, 0 and 1 carried over note: $8_{10} = 10_8$
  if the result exceeds basis B (8):
  z_i (0) write down the first digit, z_{i+1} (1) is carried over
- 7 + 1 = 10 + 1 (carry over) = 11, 1 and 1 carried over
- 1 + 6 = 7 + 1 (carry over) = 10, 0 and 1 carried over
- 0 + 0 = 0 +1 (carry over) = 1
- if there are no more digits to add, the operation is over

addition

example

basis 16:

		8	0	C
+		8	F	A
	1	1	0	6

- C + A = 6, 6 and 1 carried over
  if the result exceeds basis B (8):
  z_i (6) write down the first digit, z_{i+1} (1) is carried over
- 0 + F = F + 1 (carry over) = 10, 0 and 1 carried over
- 8 + 8 = 10 + 1 (carry over) = 11, 1 and 1 carried over
- 0 + 0 = 0 +1 (carry over) = 1, 1
- if there are no more digits to add, the operation is over

addition

example

basis 2:

		1	0	1
+		1	1	1
	1	1	0	0

- 1 + 1 = 10, 0 and 1 carried over
  if the result exceeds basis B (8):
  z_i (0) write down the first digit, z_{i+1} (1) is carried over
- 0 + 1 = 1 + 1 (carry over) = 10, 0 and 1 carried over
- 1 + 1 = 10 + 1 (carry over) = 11, 1 and 10 carried over
- 0 + 0 = 0 + 10 (carry over) = 10, 0 and 1 carried over
- if there are no more digits to add, the operation is over

subtraction

example

basis 10:

$$\begin{array}{r} 406 \\ -\ 118 \\ \hline 288 \end{array}$$

- 6 - 8 does not work because 6 < 8, so we borrow from i+1 10 (the basis): 16 - 8 = 8, 8 and 1 carried over
- add the carry over (1) to the **subtrahend** (1 + 1 carry over) -> 2 + **8** = 10, 8 and 1 carried
- add the carry over (1) to the subtrahend (1 + 1 carry over) -> 2 + **2** = 4, 2
- if there are no more digits to subtract, the operation is over

subtraction

example

basis 8:

$$
\begin{array}{cccc}
 & 4 & 3 & 7 \\
- & 1 & 5 & 6 \\
\hline
 & 2 & 6 & 1
\end{array}
$$

- 7 - 6 does work because 7 > 6,
  -> 7 - 6 = 1, 1
- 3 - 5 does not work because 3 < 5,
  so we borrow 10 (the basis) from i+1:
  3 + 10 = 13 note: $8_{10} = 10_{8}$
  -> 5 + **6** = 13, 6 and 1 carried over
- add the carry over (1) to the subtrahend (1 + 1 carry over)
  -> 2 + **2** = 4, 2
- if there are no more digits to subtract, the operation is over

subtraction

example

basis 16:

$$\begin{array}{cccc} & F & 8 & 3 \\ - & 1 & E & 1 \\ \hline & D & A & 2 \end{array}$$

- 3 - 1 does work because 3 > 1,
  -> 3 - 1 = 1
- 8 - E does not work because 8 < E,
  so we borrow 10 (the basis) from i+1:
  8 + 10 = 18 note: $16_{10} = 10_{16}$
  -> E + **A** = 18, A and 1 carried over
- add the carry over (1) to the
  subtrahend (1 + 1 carry over)
  -> 2 + **D** = F, D
- if there are no more digits to subtract,
  the operation is over

subtraction

example

basis 2:

$$
\begin{array}{rcccc}
 & 1 & 1 & 0 & 1 \\
- & & 1 & 1 & 1 \\
\hline
 & 0 & 1 & 1 & 0
\end{array}
$$

1 - 1 does work because 1 ≥ 1,
-> 1 - 1 = 0, 0

0 - 1 does not work because 0 < 1,
so we borrow 10 (the basis) from i+1:
0 + 10 = 10 note: $2_{10} = 10_2$
-> 1 + **1** = 10, 1 and 1 carried over

add the carry over (1) to the subtrahend, 1 + 1 (carry over)= 10
1 - 10 does not work because 1 < 10,
so we borrow 10 (the basis) from i+1:
1 + 10 = 11 note: $2_{10} = 10_2$
-> 10 + **1** = 11, 1 and 1 carried over

add the carry over (1) to the subtrahend, 0 + 1 (carry over) = 1
-> 1 + **0** = 1, 0

combination of mathematical operation

example

basis 8:

$$
\begin{array}{cccccc}
 & 5 & 3 & 7 & 2 & 0 \\
- & & 5 & 2 & 0 & 1 \\
- & & 1 & 2 & 3 & 4 \\
- & & 1 & 1 & 0 & 0 \\
\hline
 & 4 & 4 & 1 & 6 & 3
\end{array}
$$

add

subtract

multiplication

example

basis 10:

```
  4 5 6 7 * 3 4 5
-----------------
1 3 7 0 1
  1 8 2 6 8
    2 2 8 3 5
-----------------
1 5 7 5 6 1 5
```

- multiply the digit with the lowest index from the multiplicator with the digit with the highest index from the multiplicand
- write down the least significant number and carry the following digits of the partial result over to i+1 (digit 2)
- add the carry over to the result of the next multiplication and carry on until -> i = n
- when all partial multiplications with the highest digit of the multiplicand are done proceed with the next digit of the multiplicand as in step one. Move the partial results to the next row and one digit to the right
- sum the results up

multiplication

example

basis 8:

```
  4 5 6 7 * 3 4 5
-----------------
1 6 1 4 5
  2 2 7 3 4
    2 7 5 2 3
-----------------
2 0 7 3 5 6 3
```

- analog to the multiplication with basis 10
- you can use the modulo division
- example for modulo division basis 10 (modulo 8)

  1. row:

  7 * 3 = 21

  21 % 8 = **5**, carry over: 2

  6 * 3 = 18 (+2 carry over) = 20

  20 % 8 = **4**, carry over: 2

  5 * 3 = 15 (+2 carry over) = 17

  17 % 8 = **1**, carry over: 2

  4 * 3 = 12 (+2 carry over) = 14

  14 % 8 = **6**, carry over: 1

  0 * 3 = 0 (+ 1 carry over) = 1

  1 % 8 = **1**

division

example

basis 10:

6497 : 89 = 73
267
0

- define a block for the first digits of the dividend (from left to the right), which can be divided (possible value for the quotient: > 0 and < basis)
- estimate the result of the division (block divided by the quotient)
- multiply the divisor with the estimated result (7) and subtract it from the block -> carry over
- if the carry over > than the divisor, increase the estimation of the last division.
  if the result of the division is negativ, decrease the estimation of the last division
- if the carry over is > 0 and < divisor add the next digit from the dividend to the block (right side) -> continue

division

example

Basis 8:

$$\underbrace{1\ 5}\ 1 : 7 = 1\ 7$$

$$\ \ 6\ 1$$

$$\ \ \ \ 0$$

- analog to the division with basis 10
- estimate the result of the division (block divided by quotient)
- multiply the divisor with the estimated result (1) and subtract it from the block -> carry over
  $15_8 - (1_8 * 7_8) = 6_8$
- next digit one row down
- estimate the result of the division (block divided by quotient)
- multiply the divisor with the estimated result (7) and subtract it from the block -> carry over
  $61_8 - (7_8 * 7_8) = 0_8$

division

example

basis 16:

```
E 6 : A = 1 7
4 6
  0
```

- analog to the division with basis 10
- estimate the result of the division (block divided by quotient)
- multiply the divisor with the estimated result (1) and subtract it from the block -> carry over
  $E_{16} - (1_{16} * A_{16}) = 4_{16}$
- next digit one row down
- estimate the result of the division (block divided by quotient)
- multiply the divisor with the estimated result (7) and subtract it from the block -> carry over
  $46_{16} - (A_{16} * 7_{16}) = 0_{16}$

addition with basis B and a fractional part

- put the commas in one vertical line and fill the empty digits with 0.

basis 8:

3	5	7	.	1	2
	7	7	.	7	
4	5	7	.	0	2

by Mag. Harald Zumpf

subtraction with basis B and a fractional part

- put the commas in one vertical line and fill the empty digits with zero

basis 8:

$$
\begin{array}{rrrcrr}
3 & 5 & 7 & . & 1 & 2 \\
 & 7 & 7 & . & 7 & \\
\hline
2 & 5 & 7 & . & 2 & 2
\end{array}
$$

multiplication with basis B and a fractional part

- add the decimal places of both numbers (result x)
- result: move the comma by x decimal places to the left

basis 10:

```
   4 5 6,7 * 3,4 5
  ----------------
 1 3 7 0 1
   1 8 2 6 8
     2 2 8 3 5
  ----------------
 1 5 7 5,6 1 5
```


by Mag. Harald Zumpf

SPENGERGASSE
ausbildung mit zukunft

division with basis B and a fractional part

- every decimal place of the dividend moves the comma of the result one step to the left
- every decimal place of the divisor moves the comma of the result one step to the right
- fill empty places with 0

Basis 10:

6 4 9,7 : 8 9 = 7,3	6 4 9 7 : 8,9 = 7 3 0,
2 6 7	2 6 7
0	0

conversion of numbers to the decimal system

- every denominational number system can be converted to the decimal system
- for the conversion of a value with a fractional part we have to split up the value into a integral value (Z_{gz}) and a fractional value (Z_{gb})
- example: 456,32 -> Z_{gz} = 456 and Z_{gb} = 0,32

Z_{gz} Z_{gb}

conversion of the integral value with the Horner-algorithm to the decimal system

- every number in the denominational system can be defined by: the sum of: each digit with the index i * basis$^{\text{index i}}$

- we can use this fact with the Horner-algorithm for the integral part of a value:

$$4567_{10} = 4*10^3 + 5*10^2 + 6*10^1 + 7*10^0$$

$$4567_8 = 4*8^3 + 5*8^2 + 6*8^1 + 7*8^0 =$$
$$2048 + 320 + 48 + 7 = 2423_{10}$$

$$1221_3 = 1*3^3 + 2*3^2 + 2*3^1 + 1*3^0 =$$
$$27 + 18 + 6 + 1 = 52_{10}$$

conversion of the fractional value with the Horner-algorithm to the decimal system

- to convert the fractional part of the given value we have to multiply each digit of the fractional part (starting from the right) with the basis (to the power of the index i) of the target numeral. The sum of these result gives us the partial result Z'
- Z' has to be divided by the target basis (to the power of the highest index i+1)

$$0{,}4567_8 = 7*8^0 + 6*8^1 + 5*8^2 + 4*8^3 =$$

$$7 + 48 + 320 + 2048 = 2423 = Z'$$

$$Z_{gb} = (Z' : 4096) = 0{,}59155...$$

conversion of values from the decimal system to any given numeral system

- every value from the decimal system can be converted to any other numeral system
- for the conversion of a value with a fractional part we have to split up the value into a integral value (Z_{gz}) and a fractional value (Z_{gb})
- example: 456,32 -> Z_{gz} = 456 und Z_{gb} = 0,32

conversion of the integral part to any numeral system

- the integral part has to be divided by the value of the target basis

  example: **<u>109</u>,7890625**

div. by target basis	integral part of result	fractional part of result	index
109 / 2	54	1	z_0
54 / 2	27	0	z_1
27 / 2	13	1	z_2
13 / 2	6	1	z_3
6 / 2	3	0	z_4
3 / 2	1	1	z_5
1 / 2	0	1	z_6

write down the fractional part of each result from bottom to top: Z_{gz} = **1101101**

by Mag. Harald Zumpf

conversion of the fractional part to any numeral system

- the fractional part has to be multiplied with the value of the target basis
  example: **109,<u>7890625</u>**

multiplication with target basis	result	integral part	index
0,7890625	1,578125	1	Z_{-1}
0,578125	1,15625	1	Z_{-2}
0,15625	0,3125	0	Z_{-3}
0,3125	0,625	0	Z_{-4}
0,625	1,25	1	Z_{-5}
0,25	0,5	0	Z_{-6}
0,5	1	1	Z_{-7}

write down the integral part of each result from top to bottom: Z_{gb} = **1100101**

direct conversion

- some numeral systems can be converted directly
- each digit of the hexadecimal system is represented by 4 digits of the binary system
- each digit of the octal system is represented by 3 digits of the binary system
- when converting values with this method empty spaces have to be filled with 0

direct conversion

example

- direct conversion from the hexadecimal system to the binary system
- works for the fractional part too

3				A				F				8			
0	0	1	1	1	0	1	0	1	1	1	1	1	0	0	0

direct conversion

example

- direct conversion from the octal system to the binary system
- works for the fractional part too

7			0			5			2			1		
1	1	1	0	0	0	1	0	1	0	1	0	0	0	1

SPENGERGASSE
ausbildung mit zukunft

fixed point representation
data types

- binary representation of values
- representation with n bits of memory = 2^n unique values

data type	memory usage (bit)	possible unique values
boolean	1	2
char / byte	8	256
word / short / int	16	65536
dword / long	32	4.294.967.296
int64 / long long	64	2^{64}
int128	128	2^{128}

by Mag. Harald Zumpf

fixed point representation
signed / unsigned data types

- signed = usage of a leading sign
- unsigned = **no** usage of a leading sign
  -> representation of positive values
- if only positive values have to be represented the unsigned data types give us the doubled amount of unique values.
  example data types: unsigned int, unsigned word, unsigned long,...

fixed point representation notation

Vz	g	g	g	g	g	g	g	g	g	g	d	d	d	d	d

leading sign — integral part/digits — fractional part/digits

- **B** basis
- **l** length (l = Vz + g + d)
- **g** number of integral digits
- **d** number of fractional digits
- **Vz** leading sign

fixed point representation notation

example

-	0	0	0	0	1	2	3	4	8	9	2	5	0	0	0

- **B** = 10
- **l** = 16
- **g** = 10
- **d** = 5
- **Vz** = -

$$\left.\begin{array}{l} B = 10 \\ l = 16 \\ g = 10 \\ d = 5 \\ Vz = - \end{array}\right\} \quad Z = \mathbf{-123489{,}25_{10}}$$

by Mag. Harald Zumpf

fixed point representation notation

example

+	A	5	F	1	0	A	0

- **B** = 16
- **l** = 8
- **g** = 5
- **d** = 2
- **Vz** = +

Z = $\mathbf{A5F10{,}A_{16}}$

fixed point representation
binary system

- in computer systems the fixed point representation is used mostly with the binary system
- every value is coded, stored and calculated with the binary system
- with the fixed point representation we don't have any commas so it suits very well for the representation of integral values

fixed point representation binary system

example

0	1	0	1	1	1	1	1

virtual comma

- **B** = 2
- **l** = 8
- **g** = 8
- **d** = 0
- **Vz** = 0

$Z = \mathbf{1011111_2 = 95_{10}}$

fixed point representation binary system

example

0	0	0	1	0	1	0	0

virtual comma

- **B** = 2
- **l** = 8
- **g** = 8
- **d** = 0
- **Vz** = 0

Z = $\mathbf{10100_2 = 20_{10}}$

by Mag. Harald Zumpf

fixed point representation
one's complement

- the one's complement is very common operation with the fixed point representation
- every bit of a given value has to be **inverted** (1 becomes 0, 0 becomes 1)
- this method is called bit flip

fixed point representation one's complement

example

- **B** = 2
- **l** = 8
- **g** = 8
- **d** = 0
- **Vz** = 0

0	0	0	1	0	1	0	0

one's complement

1	**1**	**1**	**0**	**1**	**0**	**1**	**1**

fixed point representation two's complement

- the two's complement is done by two steps:
  1. perform the one's complement
  2. add the binary number 1 to the value
- with this method negative values are represented with computers using the fixed point representation (the most left bit flips)
- with the two's complement every given value can be represented as it's negative value and vice versa.

fixed point representation two's complement

example

- **B** = 2
- **l** = 8
- **g** = 8
- **d** = 0
- **Vz** = 0

0	0	0	1	0	1	0	0

= 20_{10}

↓ one's complement

1	**1**	**1**	**0**	**1**	**0**	**1**	**1**

two's complement ↓ **+1**

1	**1**	**1**	**0**	**1**	**1**	**0**	**0**

= -20_{10}

calculation within fixed point representation

- **Addition:** normal addition
- **Subtraction:** addition with two's complement
- **Multiplication with negative value:** multiply the positive values and perform the two's complement (result)
- **Division by negative value:** divide by positive value and perform the two's complement (result)

SPENGERGASSE
ausbildung mit zukunft

floating point representation data types

- this is a binary representation of values with a fixed-point part (**mantissa**) and a variable part (**scaling factor**)
- very large/small values can be represented
- negative values are **not** represented with the **two's complement**

data type	memory usage (Bit)	range of values
float (single precision)	32	10^{-45} - 10^{45}
double	64	10^{-308} - 10^{308}
long double	128	10^{-4966} - 10^{4966}

by Mag. Harald Zumpf

floating point representation notation

Vm	m	m	m	m	m	m	m	m	m	m	Ve	e	e	e	e

leading sign mantissa

mantissa bits

scaling factor

leading sign scaling factor

- **B** basis
- **l** length of representation (memory usage): (l = Vm + m + Ve + e)
- **Vm** leading sign of mantissa
- **m** mantissa bits
- **Ve** leading sign of scaling factor
- **e** scaling factor bits

floating point representation normalization

- every value has to be normalized for the floating point representation
- precondition for the normalized mantissa representation:
  $-1 < m \leq -0{,}1$ and **$0{,}1 \leq m < 1$**
  -> the first digit of the fractional part has to differ from 0
- the normalization is performed by a multiplication with the target basis:
  $B^{\text{number of fractional places}}$
- zB.: $238{,}25_{10}$ = $\mathbf{0{,}23825_{10} * 10^3}$ or $0{,}001101_2$ = $\mathbf{0{,}1101_2 * 2^{-2}}$
- in the following **order:** normalize, convert (to binary), normalize

floating point representation normalized representation

example

$\mathbf{Z} = 5{,}75_{10} = 101{,}11_2 = 0{,}10111_2 * 2^3$

- **B** = 2
- **l** = 10
- **m** = 5
- **e** = 3

Vm	m	m	m	m	m	Ve	e	e	e
0	1	0	1	1	1	0	0	1	1

floating point representation normalized representation

example

$$\mathbf{Z} = 11_8 = 001001_2 = 0{,}1001_2 * 2^4$$

- **B** = 2
- **l** = 10
- **m** = 5
- **e** = 3

Vm	m	m	m	m	m	Ve	e	e	e
0	1	0	0	1	0	0	1	0	0

floating point representation normalized representation - bias

- computers have to deal with very **large values**
- for this purpose we can spare out the leading sign of of the scaling factor and by the **addition** of a constant value (k-value) **to the scaling factor**
- this k-value is called **bias**
- the representation of a value with an added bias is called k-excess or excess-representation
- by sparing out the leading sign for the exponent the memory range of the exponent bit is extended by 1 bit

floating point representation normalized representation - bias

example

$\mathbf{Z} = 17{,}3_{10} = 10001{,}0100_2 = 0{,}1000101_2 * 2^5$

- **B** = 2
- **l** = 14
- **m** = 9
- **e** = 3
- **k = 2**

Ve is transformed to e, add the k-value to the scaling factor (5_{10}) = 7_{10}

Vm	e	e	e	e	m	m	m	m	m	m	m	m	m
0	0	1	1	1	1	0	0	0	1	0	1	0	0

calculation within floating point representation

- **Addition, Subtraction:** smaller scaling factor has to be adapted to larger one
- **Multiplication:** addition of scaling factors
- **Division:** subtraction of scaling factors

SPENGERGASSE
ausbildung mit zukunft

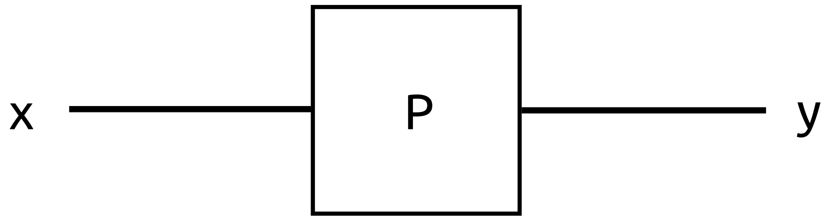
electrical circuits

- switches open or close electrical circuits.
- circuits are constructed with switches and wires to connect two points (x, y) with each other.
- a switch can be opened or closed.
  open switches are defined with the value 0 and
  closed switches with the value 1 or (L).
- the combination of multiple switches within an electrical circuit can be represented with boolean operators.

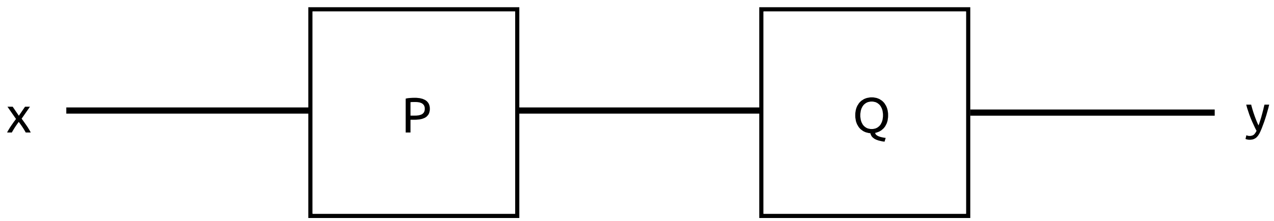
by Mag. Harald Zumpf

electrical circuits
notation

simple switch P:



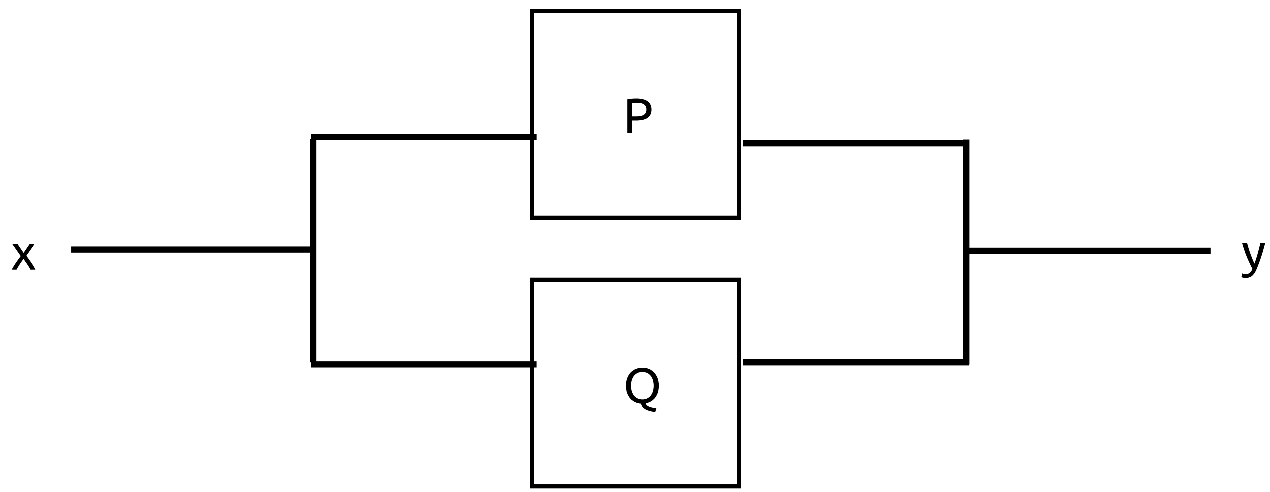
series circuit (aka. daisy chain) with the switches P and Q:



$a \wedge b$

electrical circuits notation

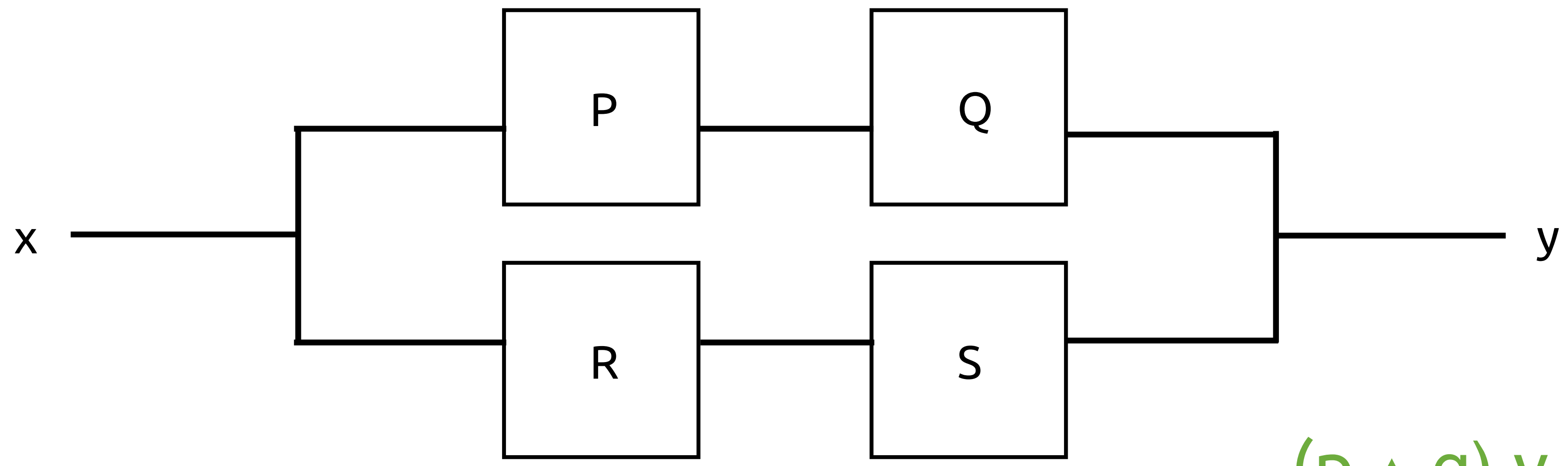
parallel circuit with the switches P and Q:



$p \vee q$

electrical circuits
notation

combination of serial and parallel circuits:

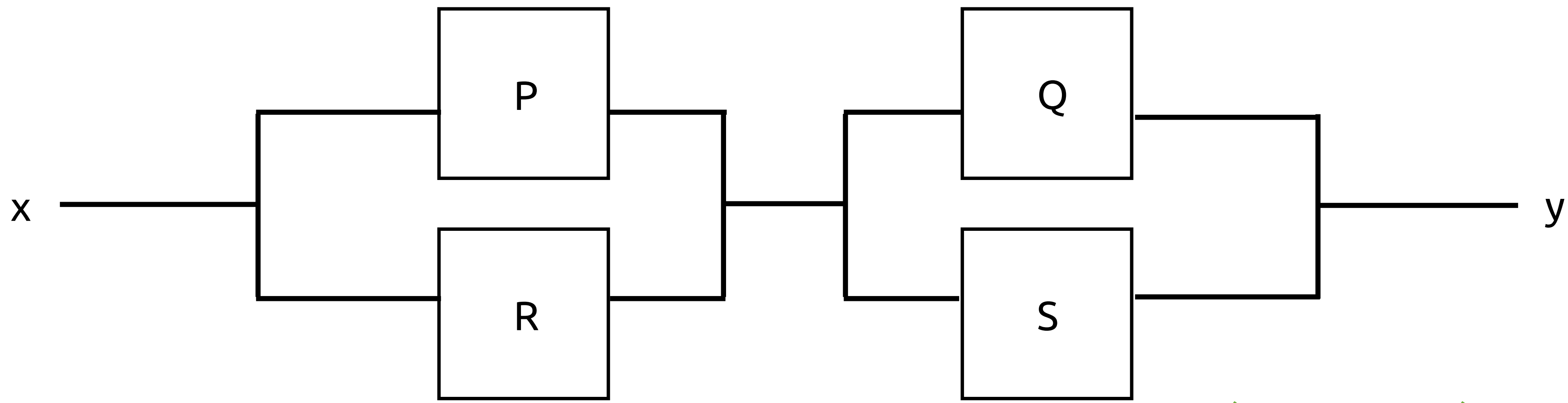


$(p \wedge q) \vee (r \wedge s)$

- notation as DNF (disjunctive normalform)

electrical circuits
notation

combination of serial and parallel circuits:



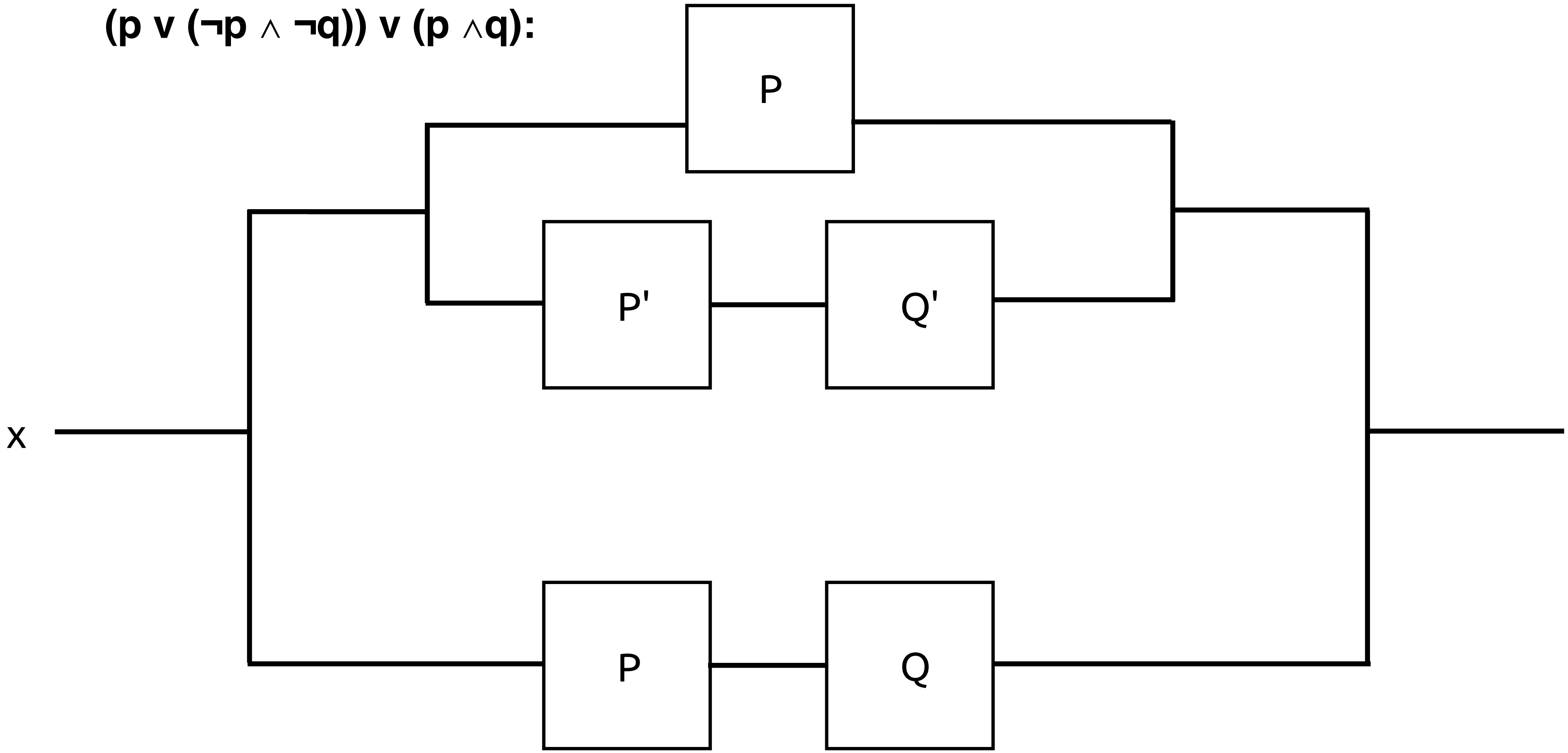
$(p \vee r) \wedge (q \vee s)$

- notation as CNF (conjunctive normalform)

electrical circuits notation

example

(p v (¬p ∧ ¬q)) v (p ∧q):



electrical circuits
link to boolean algebra

boolean algebra	electrical circuit
variable	switch
term	electrical circuit
true (t)	switch closed (L,1)
false (f)	switch opened (0)
truth table results	switching function
disjunction	parallel circuit
conjunction	serial circuit
negation	inverted switching

electrical circuits

example

- a committee (3 members) wants to perform a secret election. A bulb should light up if at least two members push a button.

- definition of three variables (p, q, r):
  p: member p agrees.
  q: member q agrees.
  r: member r agrees.

- definition of the truth table.

electrical circuits

example

truth table:

p	q	r	switching function
1	1	1	**1**
1	1	0	**1**
1	0	1	**1**
1	0	0	**0**
0	1	1	**1**
0	1	0	**0**
0	0	1	**0**
0	0	0	**0**

electrical circuits

example

- DNF (disjunctive normalform):

p	q	r	switching function	
1	1	1	**1**	$p \wedge q \wedge r$
1	1	0	**1**	$p \wedge q \wedge \neg r$
1	0	1	**1**	$p \wedge \neg q \wedge r$
1	0	0	**0**	
0	1	1	**1**	$\neg p \wedge q \wedge r$
0	1	0	**0**	
0	0	1	**0**	
0	0	0	**0**	

electrical circuits

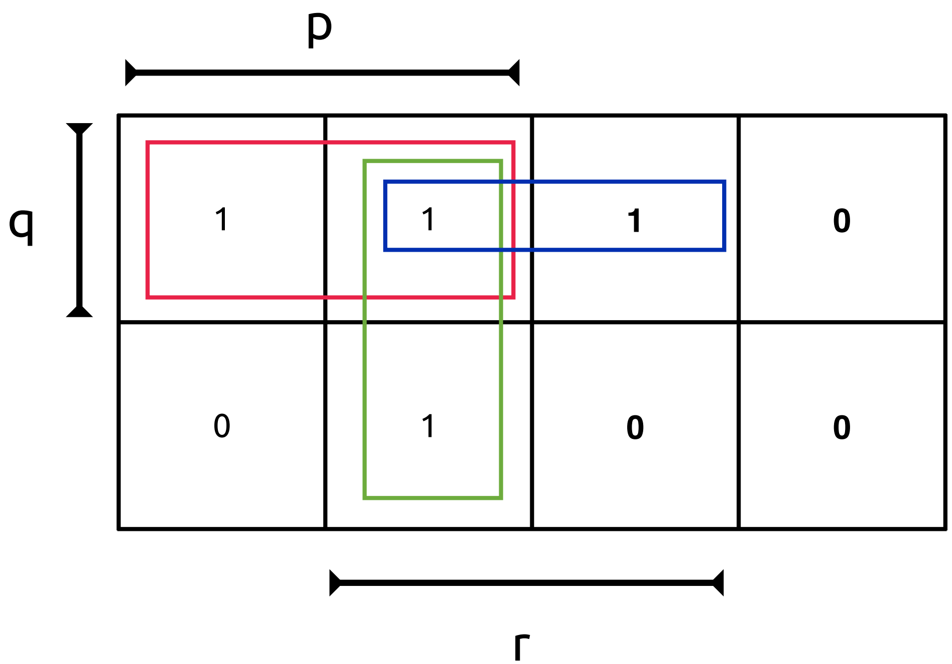
example

- DNF (disjunctive normalform):

  $(p \wedge q \wedge r)$ **v** $(p \wedge q \wedge \neg r)$ **v** $(p \wedge \neg q \wedge r)$ **v** $(\neg p \wedge q \wedge r)$

- usage of KV-diagram:

p	q	r	switching function
1	1	1	**1**
1	1	0	**1**
1	0	1	**1**
1	0	0	**0**
0	1	1	**1**
0	1	0	**0**
0	0	1	**0**
0	0	0	**0**

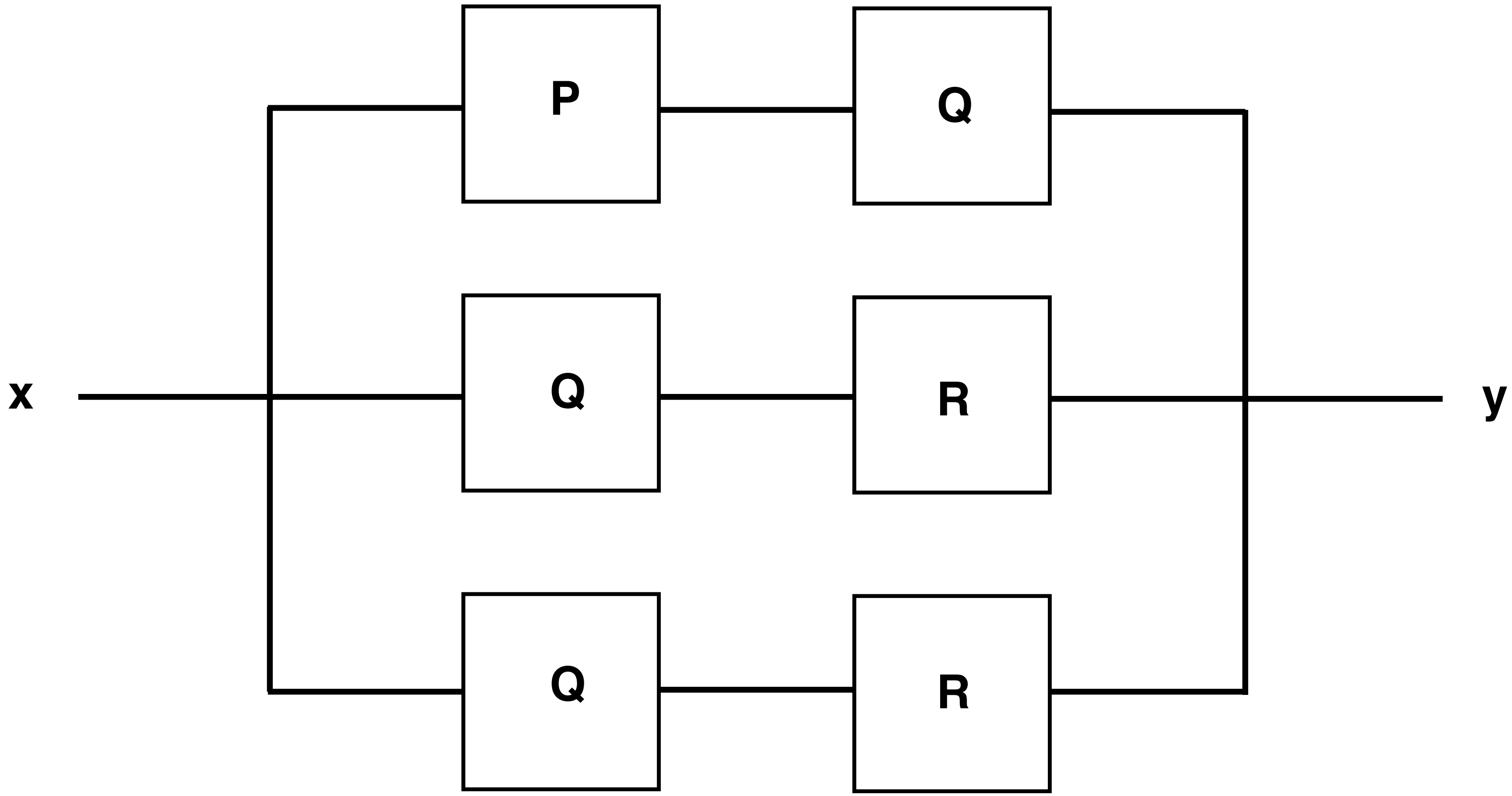


$(p \wedge q)$ **v** $(q \wedge r)$ **v** $(p \wedge r)$

electrical circuits

example

- create circuit out of result:
  $(p \land q) \lor (q \land r) \lor (q \land r)$



predicate logic

- predicate logic gives us a **deeper look** into boolean terms.

- analysis of premise and conclusion:
  All humans have parents.
  Peter is a human.
  -> Peter has parents.

Peter is a human.

subject

predicate (sentence)

predicate logic

- **subjects** are called **variables** (v, w, x, y, z).
- **predicates** are called **sentences** (A, B, C, ...).
  predicates are usually abbreviated with their beginning letters:

W	**w**orks	W(v)	v workds
C	is a **C**ity	C(w)	w is a city
R	is **r**ound	R(x)	x is round
O	is **o**n	O(y)	y is turned on

predicate logic
relations

- properties of one element are called predicates.
- predicates which define relations between several elements are called **relations:**

L	**l**ies between	L(x, y, z)	x lies between y and z
B	is **b**igger than...	B(x, y)	x is bigger than y
M	is **m**other of...	M(x, y)	x is mother of y
C	is **c**heaper than...	C(x, y)	x is cheaper than y

relations are represented by their boolean or mathematical symbol: +, -, v, ...

predicate logic
relations

- predicates are usually used to represent relations in **database systems.**
- this reduces redundancy of datasets (multiple and equal sets of data).

$$\mathbf{P}\ (x_1, x_2, x_3, \ldots, x_n)$$

predicate that links multiple datasets

n multiple datasets as:
fast name, last name, gender, ...

predicate logic
relations and usage

example

$Q(x_1, x_2, x_3, x_4)$

Q	combination of attributes
x_1	first name
x_2	last name
x_3	gender
x_4	date of birth

Q (Max, Mustermann, m, 13111980)

$R(u, v, w, x)$

R	route with **u** from **v** over **w** to **x**
u	type of plane
v	departure (airport)
w	stopover (airport)
x	destination (airport)

R (Boing747, Wien, Frankfurt, London)

predicate logic
important equivalences

$$\bigwedge_x a \wedge \bigwedge_x b \leftrightarrow \bigwedge_x (a \wedge b)$$

$$\bigwedge_x a \vee \bigwedge_x b \leftrightarrow \bigwedge_x (a \vee b)$$

$$\bigvee_x a \vee \bigvee_x b \leftrightarrow \bigvee_x (a \vee b)$$

$$\bigvee_x (a \wedge b) \leftrightarrow \bigvee_x a \wedge \bigvee_x b$$

by Mag. Harald Zumpf

quantifiers

- Quantoren erlauben es aussagenlogisch definierte Sätze durch **Generalisieren** und **Quanifizieren** noch analysierbarer zu machen.

- Man bedient sich hierbei Begriffen wie:
  Es gibt (mindestens) ein...
  Für alle Elemente gilt...
  Für kein Attribut gilt...
  etc.

- Quantoren wie manche..., einige... und viele... sind auf Grund ihrer Unbestimmtheit für die klassische Logik nicht anwendbar.

Quantoren Notation

- Zur Darstellung von aussagenlogisch analysierbaren Sätzen benötigt man bei der Verwendung von Quantoren natürlich gewisse Operatoren, die den Bool'schen Operatoren in Darstellung und Anwendung ähneln:

Bezeichnung	**Schreibweise**	**Bedeutung**
Allquantor	$\bigwedge_{x}$	Für alle x
Existenzquantor	$\bigvee_{y}$	Es gibt ein y

Quantoren
Notation in Kombination mit Prädikaten

- Der Einsatz von Quantoren eignet sich besonders für die Prädikatenlogik. Man beachte die Ähnlichkeit zu Konjunktion und Disjunktion:

$$\bigwedge_{x} P(x)$$

Für alle x gilt, dass sie die Eigenschaft P haben.
x_1 hat Eigenschaft P **und** x_2 hat Eigenschaft P **und** x_3 hat...

$$\bigvee_{y} P(y)$$

Es gibt (midestens) ein y, das die Eigenschaft P hat
y_1 hat Eigenschaft P **oder** y_2 hat Eigenschaft P **oder** y_3 hat...

Quantoren Äquivalenzen

- Die Regel von DeMorgan lässt sich auch auf Quantoren anwenden. Dadurch entstehen wichtige Äquivalenzen:

Darstellung	logisches Äquivalent
$\bigwedge_{x} P(x)$	$\neg \bigvee_{x} \neg P(x)$
$\bigvee_{x} R(x)$	$\neg \bigwedge_{x} \neg R(x)$

Quantoren Äquivalenzen

BEISPIEL

- Die Regel von DeMorgan lässt sich auch auf Quantoren anwenden. Dadurch entstehen wichtige Äquivalenzen:

Darstellung	Beispiel	Äquivalenz	Beispiel
$\bigwedge_{x} P(x)$	Alle Fische können schwimmen.	$\neg\bigvee_{x}\neg P(x)$	Es gibt keinen (einen) Fisch, der nicht schwimmen kann.
$\bigvee_{x} R(x)$	Es gibt ein Dorf mit mindestens 500 Einwohnern.	$\neg\bigwedge_{x}\neg R(x)$	Nicht alle Dörfer haben weniger als 500 Einwohner.

Quantoren
Aussagenlogische Terme

BEISPIEL

- Bei der Kombination von Prädikatenlogik und Quantoren ist oftmals die Positionierung der Negation von großer Bedeutung:

Darstellung	Beispiel
$\neg \bigwedge_{x} S(x)$	Nicht alle Wassertiere können schwimmen. (Das heißt, dass es zum Beispiel auch Krebse gibt, die nicht schwimmen können.)
$\bigwedge_{x} \neg S(x)$	Alle Wassertiere können nicht schwimmen. (Das heißt, dass es zum Beispiel auch Fische nicht schwimmen können würden.)

Quantoren
Aussagenlogische Terme

BEISPIEL

- Bei der Kombination mehrerer Quantoren ist die Positionierung des jeweiligen Quantors von Bedeutung.

Darstellung	Beispiel
K(x,y) x, y	x kennt y. Lehrer, Schüler
$\bigvee_x \bigwedge_y K(x,y)$	Ein Lehrer kennt alle Schüler.
$\bigwedge_x \bigvee_y K(x,y)$	Ein Schüler kennt alle Lehrer.

by Mag. Harald Zumpf

Quantoren
Aussagenlogische Terme

BEISPIEL

- Bei der Kombination mehrerer Quantoren ist die Positionierung des jeweiligen Quantors von Bedeutung.

Prädikat	**Zuweisung**
N(x,y,z)	x Schüler haben y-mal die Note z.

30 Schüler bekommen Zeugnisse:
Wenn auf 30 Schüler 91 Sehr gut und 125 Befriedigend entfallen, dann hat (mindestens) ein Schüler (mindestens) vier Sehr gut und (mindestens) ein Schüler (mindestens) fünf Befriedigend.

$$N(30,91,1) \wedge N(30,125,3) \rightarrow \bigvee_x \bigvee_y \bigvee_z (N(x,y,z) \wedge y>3 \wedge z=1) \wedge \bigvee_x \bigvee_y \bigvee_z (N(x,y,z) \wedge y>4 \wedge z=3)$$

Hausübung / Übung:

BEISPIEL

- Definiere einen einen aussagenlogischen Term mittels Quantoren und Prädikate für folgenden Sachverhalt:

Prädikat	Zuweisung
L(x,y)	x Hühner legen y Eier

Ein Bauer hat 100 Hühner:
Wenn 50 Hühner ein Ei legen, dann legen maximal 40 Hühner zwei Eier und 10 Hühner mindestens drei Eier.