Дискретна математика

доц. д-р Тодорка Глушкова,
Катедра „Компютърни технологии“, ФМИ

4

Логика

Съдържание

- Съждителна логика
- Синтаксис
- Семантика
- Теория на доказателствата
- Предикатна логика
- Приложения

Логика

- Математиката, за разлика от други научни дисциплини работи само с неоспорима истина.
- Една от основните цели на математиците е да определят кои твърдения са верни и кои -не.
- Изследването на съждения, начина им на конструиране, смисъла и доколко може да се докаже верността им е предмет на дисциплината "математическа логика"
- Логиката е научна дисциплина. Корените и можем да търсим в древна Гърция, а първия опит да се постави на научна основа принадлежи на Аристотел в книгата му "Аналитика".

Логика

- Логиката е необходимо средство и за математици и за компютърни специалисти. С нейна помощ един математик може да докаже верността на едно твърдение, а един компютърен специалист - че една програма работи така както се очаква за всички възможни въвеждания на данни.
- **Логически оператори** - така както операциите и релациите в математиката се използват за съставяне на по-сложни изрази, така и логическите оператори ще ни помогнат да образуваме съставни твърдения.

Логика

- Числовата алгебра използва за означаване на операции и релации символи (+,-,<,>…). Логиката също използва подобни символи, които съставляват нейния език.
- Алгебрата има свои закони - асоциативен, комутативен, дистрибутивен и др.
- Логиката има също свои закони, подобни на алгебричните.
- Съждителната логика, разглеждана като аналог на алгебрата се нарича “Логическа алгебра” или “Съждително смятане”.

Съждителна логика

- Най-простият вариант на математическата логика
  - Общо подмножество (най-малък общ делител) на всички останали логики.
- Предмет: атомарни съждения и връзките между тях
- Предимства:
  - Просто изграждане
  - Ясна структура
- Недостатък:
  - Недостатъчна изразителна сила
- Въпреки това, значението ѝ е огромно
  - Теоретична основа за разработване на хардуер и софтуер
  - Поведението на комбинаторни хардуерни схеми на логическо ниво може да се представи директно като логически формули

Логически съюзи и съставни съждения

<u>Съждението</u> е твърдение, за което със сигурност можем да кажем дали е вярно или не. От синтактична гледна точка то е просто изречение с подлог и сказуемо.

<u>Напр</u>.: *"Иван е човек", "5 < 8" и т.н.*

- Ако са дадени две съждения можем да ги свържем като използваме различни съюзи в съставно съждение.

Логически съюзи и съставни съждения

Съюзите за връзка могат да бъдат: “И”, “ИЛИ”, “АКО…ТО”.

- <u>*Напр*</u>.: *За простите съждения- “4<7”; “8 е четно число”, можем да образуваме съставните съждения:*
- *“4<7 и 8 е четно число”;*
- *“4<7 или 8 е четно число”;*
- *“Ако 4<7, то 8 е четно число”.*

За всяко едно от получените съставни съждения можем да кажем дали е вярно или не, т.е. отново получаваме съждение

Следователно тези съюзи са затворени оператори в множеството на всички съждения

Логически съюзи и съставни съждения

За всяко съждение можем да получим неговото отрицание с помощта на частицата “не”.

- *<u>Пример:</u>* *“4 не е < 7”*

Отрицанието е унарна логическа операция.

- **<u>Въпрос:</u>** Как можем да установим верността на съставните съждения?
- **<u>Отговор:</u>** С верностна таблица!

Логически оператори

- Нека е дадено съждението P. Тогава ***<u>отрицанието</u>*** на P ще записваме ⌉ P и ще казваме, че ⌉ P е вярно, когато P е невярно и обратно.
- Верностната таблица на отрицанието е:

P	**⌉P**
T	F
F	T

T – истина

F – лъжа

Логически оператори

- *<u>Пример:</u> P=" 4<7" е вярно, защото ˥P ="4 не е < 7" е грешно.*

- <u>Забележка:</u> Отрицанието ими аналог в алгебрата – знакът "-".

Конюнкция

- ***Конюнкция :*** Нека са дадени две съждения P и Q. Конюнкция на двете съждения P $\wedge$ Q е съждение, което е вярно само когато и P и Q са верни.
- Верностната таблица:

P	**Q**	**P $\wedge$ Q**
T	T	T
T	F	F
F	T	F
F	F	F

Конюнкция

- Конюнкцията има за аналог в говоримия език съюза “и”, а в алгебрата – умножението, поради което се нарича още логическо умножение.

- *<u>Пример:</u> Съждението "(4<7) $\wedge$ (8 е четно число)" е вярно съгласно първи ред от верностната таблица.*

Дисюнкция

- ***Дисюнкция:*** Нека са дадени две съждения P и Q. Дисюнкцията на тези две съждения $P \vee Q$ е съждение, което е вярно, когато поне едно от двете дадени съждения е вярно и грешно, ако и двете са грешни.
- Верностната таблица:

P	**Q**	**P $\vee$ Q**
T	T	T
T	F	T
F	T	T
F	F	F

Дисюнкция

- Дисюнкцията има за аналог в говоримия език съюза “или”, а в алгебрата – събирането, поради което се нарича още логическо събиране.
- *<u>Пример:</u> "(4<7) $\vee$ (8 е не четно число)" е вярно поради втори ред на верностната таблица.*
- След като дефинирахме тези три логически оператора, можем да ги прилагаме в различни комбинации и да получаваме съставни съждения. Тяхната верностна стойност можем да получим чрез верностна таблица.
- *<u>Например:</u> Ако P,Q и R са съждения да получим верностните стойности на съставното съждение: $R \wedge \neg(P \wedge Q)$.*

P	**Q**	**R**	**P ∧ Q**	**˥(P ∧ Q)**	**R ∧˥(P∧Q)**
T	T	T	T	F	F
T	T	F	T	F	F
T	F	T	F	T	T
T	F	F	F	T	F
F	T	T	F	T	T
F	T	F	F	T	F
F	F	T	F	T	T
F	F	F	F	T	F

Импликация

- ***Импликация:*** Ако P и Q са две съждения импликация $P \rightarrow Q$ наричаме съждението, което е грешно, само когато P е вярно, а Q- невярно и вярно във всички останали случаи.
- Ще наричаме P-хипотеза, а Q- заключение.

В говоримия език импликацията се замества с "ако...тогава".

В примера: "(4<7) $\rightarrow$ (8 е четно число)" е вярно, съгласно първи ред от верностната таблица.

P	**Q**	**$P \rightarrow Q$**
T	T	T
T	F	F
F	T	T
F	F	T

Импликация

- В разговорния език можем да го кажем и по друг начин:
- ✓ “ Ако P, то Q”;
- ✓ “P само ако Q”;
- ✓ ” P е достатъчно условие за Q”;
- ✓ ”Q е необходимо условие за P”.

- ***<u>Пример:</u>*** *P=“Вън вали.”; Q=“На небето има облаци.” Тогава:*
- ✓ *“Ако вън вали, на небето има облаци.”*
- ✓ *“Вън вали, само ако на небето има облаци.”*
- ✓ *Вън да вали е достатъчно условие за това на небето да има облаци.*
- ✓ *На небето да има облаци е необходимо условие вън да вали.*

Импликация

- Ще отбележим, че ако Q е вярно съждение, импликацията **P** $\rightarrow$ **Q** е винаги истина.

(напр. X<7 $\rightarrow$2+2=4 е вярно независимо от стойността на X.)

- Също, ако при дисюнкцията и конюнкцията реда на записване на съжденията е без значение ($P \wedge Q, Q \vee P$), то при импликацията той е много важен. Съждението **P** $\rightarrow$ **Q** не е еквивалентно на **Q** $\rightarrow$ **P.**
- В горния пример съждението “Ако вън вали, то на небето има облаци” не е еквивалентно на “Ако на небето има облаци, то вън вали”, тъй като денят може да е просто облачен.
- Когато, обаче, и двете съждения **P** $\rightarrow$ **Q** и **Q** $\rightarrow$ **P** са едновременно верни, тогава казваме, че съжденията P и Q са <u>еквивалентни</u> и записваме **P** $\leftrightarrow$ **Q.**

Еквиваленция

- ***Еквиваленцията*** **P $\leftrightarrow$ Q** е вярна, когато P и Q получават еднакви верностни стойности и невярна в останалите случаи.
- Верностна таблица:

P	**Q**	**P $\leftrightarrow$ Q**
T	T	T
T	F	F
F	T	F
F	F	T

Еквиваленция

- В математиката еквивалентността се изговаря като “тогава и само тогава, когато” или като “P е необходимо и достатъчно условие за Q”.

- ***<u>Например:</u>*** *Триъгълникът е правоъгълен тогава и само тогава, когато сборът от квадратите на катетите е равен на квадрата на хипотенузата.*

Изключващо „Или“- XOR

- ***Изключващото „Или"*** **P ⊕ Q** е вярна, само когато P и Q получават различни верностни стойности и невярна в останалите случаи.
- Верностна таблица:Нарича се още „сума по модул 2“ или „изключващо ИЛИ“.
- Означава се с:

$$f=P\oplus Q$$

P	**Q**	**P ⊕ Q**
T	T	F
T	F	T
F	T	T
F	F	F

От верностната таблица се вижда, че XOR е противоположен на еквиваленцията и затова понякога се нарича антиеквиваленция и се бележи с $\nleftrightarrow$

Побитови операции или **Шифри срещу Кодове**

- За повечето хора това е все едно да ги попиташ каква е разликата между смесване и разбъркване.
- Кодът е свързване на някаква смислена единица – като дума, изречение или фраза – с нещо друго – обикновено по-кратка група символи. Например можем да направим код, в който думата "apple" (ябълка) е записана като 67.
- *Кодовата книга* е списък от тези връзки.
- За да създаваме и разчитаме кодове се нуждаем от кодова книга или кодова формула.
- Тогава какво е шифър?

Побитови операции или **Шифри срещу Кодове**

- Най-важното е, че шифрите не включват значение.
- Те са механични операции, познати като алгоритми, които се прилагат върху отделни букви или малки групи от тях.
- Например в Цезаровия шифър всяка буква от азбуката се превръща в друга буква. Например A→D, B→E и C→F, според някакво изместване, в този случай четири.
- Този вид шифър е познат като **шифър с отместване**.
- В този случай не ни трябва кодова книга.

Побитови операции или **Шифри срещу Кодове**

- Вместо това следваме поредица от инструкции, познати още като **алгоритъм**, където преместваме всяка буква с някакъв брой позиции.
- Алгоритъмът изисква някаква споделена информация, позната като **ключ**.
- В горния пример, в който A→D, ключът е четири. Този споделен ключ се изисква от двете страни, за да **кодират** съобщения: HELLO → KHOOR, както и да **декодират** съобщения: KHOOR→HELLO.

Побитови операции или **Шифри срещу Кодове**

- Да се върнем на въпроса: Каква е разликата между код и шифър?
- Кодовете се отнасят към ***семантичното*** значение, докато шифрите оперират със ***синтаксиса***, символите.
- Кодът се запазва в кодова книга, докато шифрите превръщат отделните символи един в друг според някакъв алгоритъм.

Побитови операции или **Шифри срещу Кодове**

- Най-добрият шифър с отместване е шифърът с еднократен код, при който всеки символ се шифрова с различно случайно отместване.
- Той се осъществява чрез прилагането на последователност от случайни замествания с дължина равна на дължината на съобщението.
- Този шифър е трудно пробиваем, тъй като всяка комбинация е равновероятна на останалите, а броят комбинации нараства експоненциално.

Побитови операции или **Шифри срещу Кодове**

- Например при шифроване на думата КОД с възможно отместване 30 (колкото са буквите в българския език) получаваме

К О Д

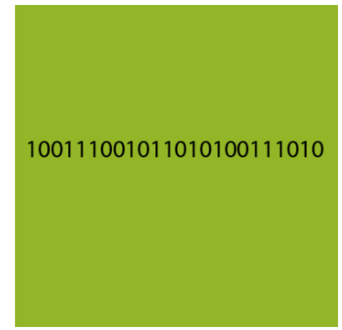
30* 30* 30=27000 различни равновероятни трибуквени комбинации.

Побитови операции или **Шифри срещу Кодове**

- Причината за това е използването на побитовите побитовите операции **AND** (и), **OR** (или) и **XOR** (изключващо или).
- И най-вече, трябва да разберем защо XOR трябва да се използва, когато се използва шифър с еднократен код на компютри.
- **Побитово** просто означава, че работим с отделни битове или **двоични числа**.
- Във всяка съвременна компютъризирана схема за критиране ние представяме символите с 0 и 1.

Криптиране на цветове

- Всеки RGB цвят е комбинация от три осембитови числа. Например: (R) = 156, (G) = 161 и (B) = 58.
- Ако изразим числата двоично, получаваме: R=10011100, G=10110101, B=00111010.
- Можем да ги представим заедно така: 100111001011010100111010



- Сега генерираме отместване с произволно число със същата дължина (напр. като хвърляме 24 пъти монета)
- За да извършим кодирането с еднократен случаен ключ, трябва да изберем правилната операция, така че получената последователност да може да е който и да е цвят с еднаква вероятност.
- Нека да разгледаме различните операции: **AND**, **OR**, **XOR**.

Криптиране на цветове

Логическо «И» (AND)

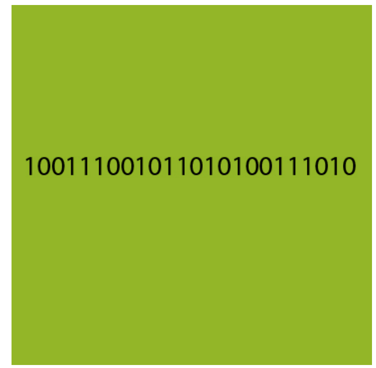
неговата таблица на истинност:

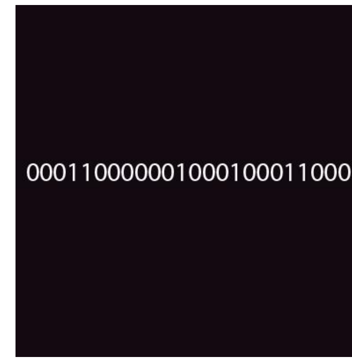
0 AND **0** = **0**

0 AND **1** = **0**

1 AND **0** = **0**

1 AND **1** = **1**





Нека опитаме:

100111001011010100111010 **AND** 010110100001101111011000 = **000110000001000100011000**

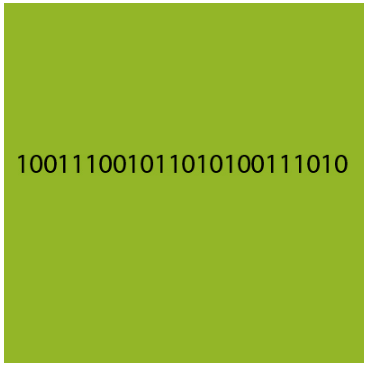
Полученият цвят е много тъмно лилаво.

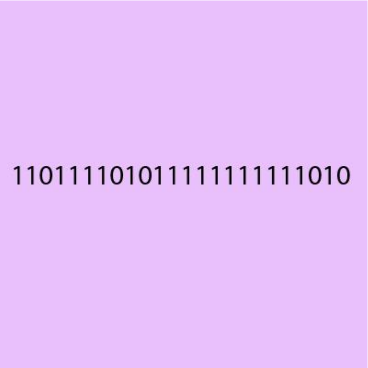
Когато прилагаме операцията **AND**, резултът **не може да е по-голям от него**.

Криптиране на цветове

Логическо «Или» (OR)

0 OR **0** = **0**
0 OR **1** = **1**
1 OR **0** = **1**
1 OR **1** = **1**





Нека опитаме:

100111001011010100111010 **OR** 010110100001101111011000
= **110111101011111111111010**

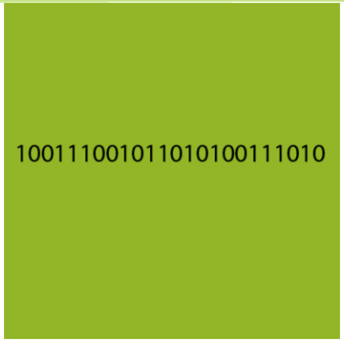
Резултатът е светло лилаво.

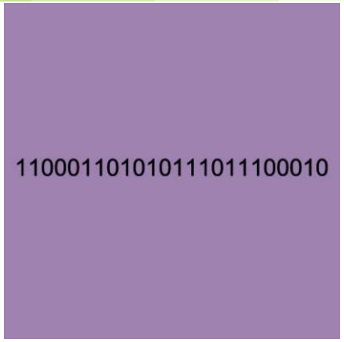
Когато прилагаме операцията **OR**, получената последователност **не може да е по-малка**.

Криптиране на цветове

Изключващо «Или» (XOR)

0 XOR **0** = **0**
0 XOR **1** = **1**
1 XOR **0** = **1**
1 XOR **1** = **0**





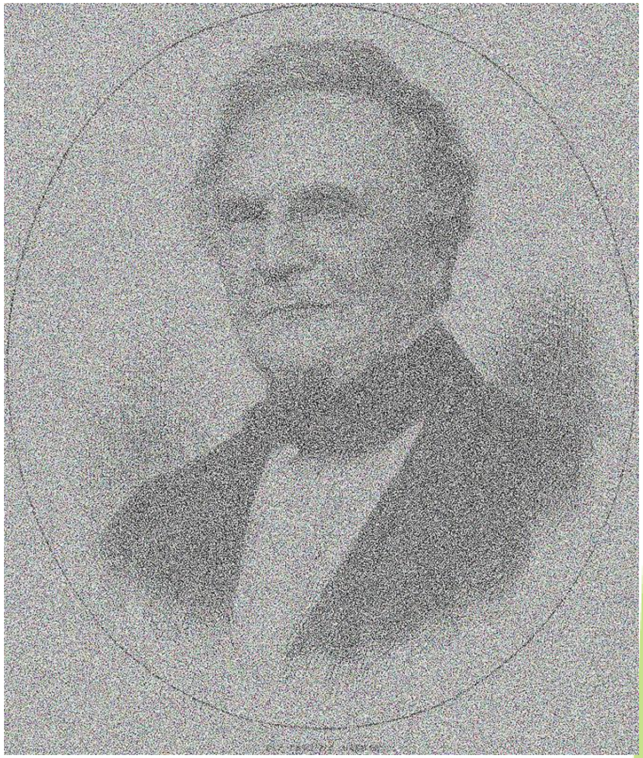
Нека опитаме:

100111001011010100111010 **XOR** 010110100001101111011000 = 110001101010111011100010

- Резултатът е малко по-тъмно лилаво в сравнение с OR.
- Когато използваме операцията XOR, **получената последователност може да бъде всяка възможна**
- Ако вземем някакъв криптиран цвят, всичко което знаем е, че първоначалният цвят “е еднакво вероятно да бъде всеки цвят” и е необходимо <u>сляпо отгатване</u> (1/16 милиона).

Попикселно криптиране на изображения

AND

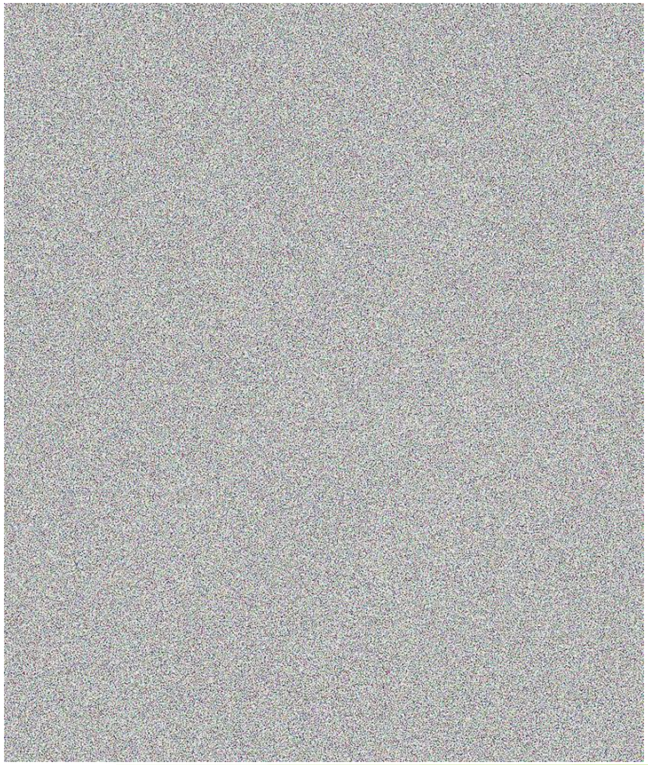


Попикселно криптиране на изображения

OR

Попикселно криптиране на изображения

XOR



Тавтология и еквивалентност

Някои съставни съждения са винаги верни, а други – винаги неверни независимо от верностните стойности на съставляващите ги съждения.

- ***<u>Например:</u>*** ***$P \vee \neg P$*** *е винаги вярно твърдение, докато* ***$P \wedge \neg P$*** *е винаги невярно.*
- *Това можем да видим от верностните им таблици:*

P	**$\neg P$**	**$P \vee \neg P$**
T	F	T
F	T	T

P	**$\neg P$**	**$P \wedge \neg P$**
T	F	F
F	T	F

Тавтология и противоречие

- ***<u>Тавтология </u>***: Всяко съждение, което е винаги вярно, независимо от верностните стойности на съставляващите го съждения.

- ***<u>Противоречие:</u>*** Съждение, което е винаги невярно.(Второто съждение от горния пример.)
- Можем лесно да ги разпознаем, ако във верностната таблица получим само T (тавтология) или само F (противоречие).

Пример: Да докажем, че $\neg(P \vee Q) \rightarrow \neg P \wedge \neg Q$ е тавтология.

P	Q	$\neg P$	$\neg Q$	$P \vee Q$	$\neg P \wedge \neg Q$	$\neg(P \vee Q)$	$\neg(P \vee Q) \rightarrow \neg P \wedge \neg Q$
T	T	F	F	T	F	F	**T**
T	F	F	T	T	F	F	**T**
F	T	T	F	T	F	F	**T**
F	F	T	T	F	T	T	**T**

Тавтология и еквивалентност

- Нека S_1 и S_2 са две съждения. Казваме, че те са еквивалентни, когато двете колони във верностната таблица, в които те получават стойностите си са еднакви.
- В примера това са колоните за $\neg P \wedge \neg Q$ **и** $\neg(P \vee Q)$. Тогава можем да кажем, че:

$$\neg P \wedge \neg Q \Leftrightarrow \neg(P \vee Q)$$

е <u>еквивалентност</u>.

Логически твърдения

- В математическата логика съществуват два начина за доказване на логически еквивалентности:
  - дедуктивен
  - манипулативен (с верностна таблица).
- Често манипулативния подход води до бързи резултати, но при верностни таблици с много логически променливи, този подход е практически неприложим.
- Затова ще въведем някои базови еквивалентности, с чиято помощ чрез дедуктивни методи ще можем да докажем всяка друга еквивалентност.

ТЕОРЕМА 1. **Ако P, Q и R са твърдения, T е вярно твърдение, а F е грешно, следните двойки твърдения са логически еквивалентни:**

1) Комутативни закони:

$$P \vee Q \Leftrightarrow Q \vee P$$

$$P \wedge Q \Leftrightarrow Q \wedge P$$

2) Асоциативни закони:

$$P \vee (Q \vee R) \Leftrightarrow (P \vee Q) \vee R$$

$$P \wedge (Q \wedge R) \Leftrightarrow (P \wedge Q) \wedge R$$

3) Дистрибутивни закони:

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$$

$$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$$

4) Комплиментарни закони:

$$P \vee \rceil P \Leftrightarrow T$$

$$P \wedge \rceil P \Leftrightarrow F$$

5) Закони на идентитета:

$$P \wedge T \Leftrightarrow P$$

$$P \vee F \Leftrightarrow P$$

6) Доминантни закони:

$$P \vee T \Leftrightarrow T$$

$$P \wedge F \Leftrightarrow F$$

7) за идемпотентност:

$$P \wedge P \Leftrightarrow P$$

$$P \vee P \Leftrightarrow P$$

8) Закон за двойното отрицание:

$$\neg(\neg P) = P$$

9) Закони на Де Морган:

$$\neg(P \vee Q) \Leftrightarrow \neg P \wedge \neg Q$$

$$\neg(P \wedge Q) \Leftrightarrow \neg P \vee \neg Q$$

10) Закон за импликацията: $P \rightarrow Q \Leftrightarrow \rceil P \vee Q$

11) Закон за отрицание на импликацията:

$$\rceil(P \rightarrow Q) \Leftrightarrow \rceil Q \wedge P$$

12) Закон за контрапозицията:

$$P \rightarrow Q \Leftrightarrow \rceil Q \rightarrow \rceil P$$

13) Закон за еквиваленцията:

$$P \leftrightarrow Q \Leftrightarrow (P \rightarrow Q) \wedge (Q \rightarrow P)$$

Логически изводи

- Логическите изводи ни позволяват да конструираме коректни доказателства.

- <u>Дефиниция:</u> Нека S1 и S2 са съставни съждения. Казваме, че S2 следва от S1, т.е. S1 $\Rightarrow$ S2, ако за всяко разпределение на верностните стойности на съждителните променливи в S1 и S2, от верността на S1 следва верността на S2.

Пример

- **Пример:** За да докажем, че: $P \wedge (P \rightarrow Q) \Rightarrow Q$,

Конструираме верностната таблица и сравняваме колоната $P \wedge (P \rightarrow Q)$ с колона Q.

P	Q	$P \rightarrow Q$	$P \wedge (P \rightarrow Q)$
T	T	T	T
T	F	F	F
F	T	T	F
F	F	T	F

Логически изводи

Теорема 2 Следващите логически изводи са верни. P,Q,R са някакви съждения, а T и F са вярно и невярно съждение:

1) $(P \to Q)\, P \wedge \Rightarrow Q$
2) $P \Rightarrow P \vee Q$
3) $P \wedge Q \Rightarrow P$
4) $\neg P \to P \Rightarrow P$
5) $\neg P \to F \Rightarrow P$
6) $F \Rightarrow P$
7) $P \Rightarrow T$
8) $(P \to Q) \wedge (Q \to R) \Rightarrow P \to R$

Импликация и логически извод

- Да отбележим връзката между импликацията ($\rightarrow$) и логическия извод($\Rightarrow$); между еквиваленцията($\leftrightarrow$) и логическата еквивалентност($\Leftrightarrow$).

- **<u>Теорема 3.</u>** Нека P и Q са съждения. Тогава:

1) $P \Rightarrow Q$ тогава и само тогава, когато $P \rightarrow Q$;
2) $P \Leftrightarrow Q$ тогава и само тогава, когато $P \leftrightarrow Q$;

Булева алгебра

- Булевата алгебра (или алгебра на съжденията) е специална алгебрична структура, която съдържа логическите оператори И, ИЛИ, НЕ, както и множествените функции сечение, обединение, допълнение.
- Тя е дефинирана за първи път от британския математик Джордж Бул, с цел да се използват алгебрични методи в логиката. Булевата алгебра и булевите операции стоят в основата на информатиката, програмирането и функционирането на компютърните системи, тъй като компютрите са програмирани да извършват точно тези логически операции.

Булева алгебра

- Константите, с които се оперира в тази алгебра са 0 и 1
- Операторите се срещат често написани по различен начин, напр. И, ИЛИ, НЕ (англ. AND, OR, NOT); ∧, ∨, ¬;
- математиците често използват + за ИЛИ, · за И и черта над символа за НЕ.

Джордж Бул

Джордж Бул (*George Boole*)

Роден: 2 ноември 1815 г., Линкълн, Великобритания и Северна Ирландия

Починал: 8 декември 1864 г.

- Британски математик и философ
- Изобретател на Булевата алгебра, която е в основата на цялата съвременна компютърна аритметика
- Приема се като един от основателите на компютърната наука

Булева алгебра и съждителна логика

- Съждителната логика формално можем да представим в две стъпки:
  - **Синтаксис** – правила за изграждане на логически изрази (формули)
    - Формула – последователност от символи, които могат да се комбинират по определен начин
  - **Семантика** – снабдява изразите със значение
    - Определя как да бъдат интерпретирани символите в една логическа формула

Синтаксис

Дефиниция:

Множеството на логическите формули върху множеството на променливите $V = \{ A_1, A_2, \ldots \}$ се дефинира рекурсивно както следва:

- Булевите стойности 0 и 1 са формули
- Всяка променлива $A_1 \in V$ е формула
- Ако F и G са формули, тогава $(\neg F)$, $(F \wedge G)$, $(F \vee G)$, $(F \rightarrow G)$, $(F \leftrightarrow G)$, $(F \not\leftrightarrow G)$ също са формули

Синтаксис

Оператори:

- “¬": отрицание
- “∧ ": конюнкция (И-оператор)
- “∨ ": дизюнкциянюнкция (ИЛИ-оператор)
- “→ ": импликация
- “↔ ": еквивалентност
- “↮ ": антивалентност (XOR-оператор)

Формули

Атомарна формула F: Формула, която не може да бъде разлагана
В съждителната логика това е множеството на атомарните формули е идентично с множеството $V \cup \{ 0, 1 \}$

Съставна формула: Формула, която може да се разрага.
Съставните ѝ части означаваме като подформули
Използваме означението: $F \in G$, съответно $F \notin G$

Обобщения за формули:

$$\bigwedge_{i=1}^{n} F_i = F_1 \wedge \ldots \wedge F_n \qquad \bigvee_{i=1}^{n} F_i = F_1 \vee \ldots \vee F_n$$

Интерпретации

- Семантиката на формулите се определя посредством моделиращата релация $\models$
- За да можем да я дефинираме формално първо трябва да въведем понятието за интерпретация

Дефиниция:

Нека са дадени:
- F е логическа формула
- $A_1, \ldots, A_n$ променливи, съдържащи се във F

Всяка I: $\{ A_1, A_2, \ldots \} \rightarrow \{ 0, 1 \}$ се нарича интерпретация на F

Пример

$(\neg A \to \neg B) \wedge ((B \to A) \vee (A \to B))$

1

$A \to 0$
$B \to 0$

$(\neg 0 \to \neg 0) \wedge ((0 \to 0) \vee (0 \to 0))$

интерпретация

Пример

$(\neg A \to \neg B) \wedge ((B \to A) \vee (A \to B))$

$A \to 1$
$B \to 0$

$(\neg 1 \to \neg 0) \wedge ((0 \to 1) \vee (1 \to 0))$

интерпретация

2

Пример

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

3

$A \rightarrow 0$
$B \rightarrow 1$

$(\neg 0 \rightarrow \neg 1) \wedge ((1 \rightarrow 0) \vee (0 \rightarrow 1))$

интерпретация

Пример

$(\neg A \to \neg B) \wedge ((B \to A) \vee (A \to B))$

4

$A \to 1$
$B \to 1$

$(\neg 0 \to \neg 1) \wedge ((1 \to 0) \vee (0 \to 1))$

интерпретация

Въпрос

Ако имаме формула с n на брой променливи, колко е броят на възможните интерпретации?

$$2^n$$

Семантика

Нека F и G са логически формули, а I - интерпретация.
Семантиката се задава посредством релацията $\models$, дефинирана индуктивно върху изграждането на формулите както следва:

- $I \models 1$
- $I \not\models 0$
- $I \models A_i :\Leftrightarrow I(A_i) = 1$
- $I \models (\neg F) :\Leftrightarrow I \not\models F$
- $I \models (F \wedge G) :\Leftrightarrow I \models F$ и $I \models G$
- $I \models (F \vee G) :\Leftrightarrow I \models F$ или $I \models G$
- $I \models (F \rightarrow G) :\Leftrightarrow I \not\models F$ или $I \models G$
- $I \models (F \leftrightarrow G) :\Leftrightarrow I \models F$ тогава и само тогава, когато $I \models G$
- $I \models (F \not\leftrightarrow G) :\Leftrightarrow I \not\models (F \leftrightarrow G)$

Една интерпретация I за която $I \models F$ се нарича **модел** на F

Булеви функции

Всяка съждителна формула F с n променливи можем да разглеждаме като булева функция

$f^F: \{0,1\}^n \rightarrow \{0,1\}$, която за една интерпретация I приема стойност 1, само тогава, когато I е модел за F

Ако I присвоява на променливите $A_1, \ldots, A_n$ логическите стойности $b_1, \ldots, b_n$, тогава стойността на функцията

$$f^F(b_1, \ldots, b_n) = \begin{cases} 1, \text{ ако } I \models F \\ 0, \text{ ако } I \not\models F \end{cases}$$

Понеже дефиниционната област е дискретна, булевите функции могат да се представят като таблици

Таблици

Отрицание:

	A	¬A
0	0	1
1	1	0

Конюнкция:

	A	B	A ∧ B
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

Дизюнкция:

	A	B	A ∨ B
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

Импликация:

	A	B	A→B
0	0	0	1
1	0	1	1
2	1	0	0
3	1	1	1

Еквивалентност:

	A	B	A↔B
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	1

Антивалентност:

	A	B	A↮B
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Пример 1

изпълнима

$$F_1 = \underbrace{\underbrace{\underbrace{(\neg A \lor B)}_{G_1} \land \underbrace{(\neg B \lor C)}_{G_2}}_{G_4} \land \underbrace{(\neg C \lor A)}_{G_3}}_{F_1}$$

1 Колко модела?

A	B	C	G_1	G_2	G_3	G_4	F_1
0	0	0	1	1	1	1	1
0	0	1	1	1	0	1	0
0	1	0	1	0	1	0	0
0	1	1	1	1	0	1	0
1	0	0	0	1	1	0	0
1	0	1	0	1	1	0	0
1	1	0	1	0	1	0	0
1	1	1	1	1	1	1	1

Пример 2

общовалидна

$$\underbrace{F_2 = \underbrace{(\underbrace{(A\to B)}_{G_1} \wedge \underbrace{(B\to C)}_{G_2})}_{G_4} \to \underbrace{(A\to C)}_{G_3}}_{F_2}$$

1 Колко модела?

A	B	C	G_1	G_2	G_3	G_4	F_2
0	0	0	1	1	1	1	1
0	0	1	1	1	1	1	1
0	1	0	1	0	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	1	0	0	1
1	0	1	0	1	1	0	1
1	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1

Пример 3

неизпълнима

$$F_3 = \underbrace{\underbrace{\underbrace{(A \nleftrightarrow B)}_{G_1} \land \underbrace{(A \nleftrightarrow C)}_{G_2}}_{G_4} \land \underbrace{(B \nleftrightarrow C)}_{G_3}}_{F_3}$$

1 Колко модела?

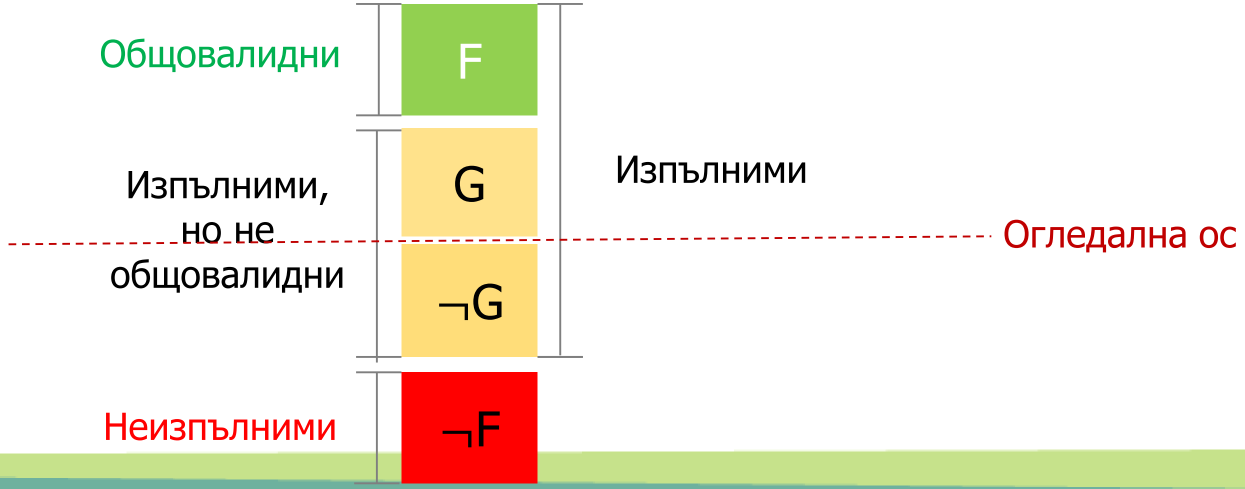
A	B	C	G_1	G_2	G_3	G_4	F_3
0	0	0	0	0	0	0	0
0	0	1	0	1	1	0	0
0	1	0	1	0	1	0	0
0	1	1	1	1	0	1	0
1	0	0	1	1	0	1	0
1	0	1	1	0	1	0	0
1	1	0	0	1	1	0	0
1	1	1	0	0	0	0	0

Видове формули

Една логическа формула F се нарича:

- Изпълнима – ако F притежава поне един модел
- Неизпълнима – ако F не притежава модел
- Общовалидна – ако ¬F е неизпълнима

Всяка общовалидна формула означаваме също като тавтология



За множества от формули

Дефиниции:

Тези три дефиниции могат да бъдат обобщени за множества от формули:

- Изпълнимо – когато съществува интерпретация I, която е модел за всяка $F_i \in M$.
  - Внимание: моделът за всички формули трябва да бъде еднакъв. Не е достатъчно всяка формула сама за себе си да бъде изпълнима.
- Неизпълнимо– когато F1, …,Fn нямат общ модел
- Общовалидно – обратното, ако всяка интерпретация е модел за елементите от M

Логически следствия

Дефиниции:

Нека $M \coloneqq \{ F_1, \ldots, F_n \}$ е множество от логически формули.
Записваме, $M \vDash G$ („от M следва G“), когато всеки модел на M е също модел на G.
Означаваме:

- $\vDash G$, за $\emptyset \vDash G$
- $F \vDash G$, за $\{ F \} \vDash G$

Освен това:

- $\vDash G$ е в сила, когато G е общовалидна
- $F \vDash G$ е в сила, когато $F \rightarrow G$ е общовалидна
- $\{ F_1, \ldots, F_n \} \vDash G$ е еквивалентна на $\{ F_2, \ldots, F_n \} \vDash F_1 \rightarrow G$

Еквивалентност

Нека F и G са логически формули.
Релацията $\equiv$ е дефинирана както следва:

- $F \equiv G :\Leftrightarrow F \models G$ и $G \models F$

Две формули F и G с $F \equiv G$ се наричат еквивалентни

Еквивалентност

Две формули F и G са еквивалентни, когато имат едни и същи модели. От математическа гледна точка "≡" е релация на еквивалентност върху множеството на съждителните формули, като притежава следните свойства:

- *Рефлексивност*: за всички формули F е в сила $F \equiv F$
- *Симетричност*: от $F \equiv G$ следва $G \equiv F$
- *Транзитивност*: от $F \equiv G$ и $G \equiv H$ следва $F \equiv H$

Освен това:

- F е само тогава общовалидна, когато $F \equiv 1$
- F е само тогава неизпълнима, когато $F \equiv 0$

Важни еквивалентности

идемпотентност

$F \wedge F \equiv F$
$F \vee F \equiv F$

неутралност

$F \wedge 1 \equiv F$
$F \vee 0 \equiv F$

Де Морган

$\neg(F \wedge G) \equiv \neg F \vee \neg G$
$\neg(F \vee G) \equiv \neg F \wedge \neg G$

комутативност

$F \wedge G \equiv G \wedge F$
$F \vee G \equiv G \vee F$

елиминация

$F \wedge 0 \equiv 0$
$F \vee 1 \equiv 1$

Двойно отрицание

$\neg\neg F \equiv F$

дистрибутивност

$F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$
$F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$

абсорбация

$F \vee (F \wedge G) \equiv F$
$F \wedge (F \vee G) \equiv F$

Значение на еквивалентностите

Значението на еквивалентностите е двояко:

- Позволява поглед върху елементарните зависимости между въведените логически оператори
- Служат като важни правила за преобразуване на съждителни изрази
  - Основа за това е субституционната теорема – позволява да заместваме подформули с еквивалентни изрази, без да влияем върху моделите на пълната формула

Пълна система от логически оператори

- Може би предизвиква учудване, че в таблицата се съдържат само съждителните елементарни оператори $\neg$, $\wedge$ и $\vee$
- Не става въпрос за ограничение

?

Всички останали могат да се свеждат до тези три. Множеството на тези три оператора се нарича пълна система от оператори

Примери

$$x \to y \equiv \neg x \lor y$$

$$x \leftrightarrow y \equiv (\neg x \land \neg y) \lor (x \land y)$$

$$\equiv (\neg x \lor y) \land (x \lor \neg y)$$

$$x \nleftrightarrow y \equiv (\neg x \land y) \lor (x \land \neg y)$$

$$\equiv (\neg x \lor \neg y) \land (x \lor y)$$

Предикатна логика

- Съществуват някои недостатъци на съждителното смятане, свързани с това, че:
  - съждителната логика изразява само логическата структура на сентенциите.
  - С нея не могат да се изразят твърдения като: “Релацията < е транзитивна” или “Всяко естествено число е равно на сумата от квадратите на 4 естествени числа.”

Предикатна логика

- Логически изводи, при които такива структури имат значение, не могат да се изразят със съждителната логика. Затова се налага да разгледаме предикатната логика.

Пример:

Всички гълъби са птици: $\forall x\,(Sx \rightarrow Mx)$

Всички птици са животни: $\forall x(Mx \rightarrow Fx)$

$\Rightarrow$ *Всички гълъби са животни:* $\forall x\,(Sx \rightarrow Fx)$

Пример

1. Сократ е човек
2. Всички хора са смъртни
3. Тогава Сократ също е смъртен

- Свойството да бъде човек е параметризирано съждение: Човек(x)
- Което в зависимост от аргумента е вярно или грешно: за индивида „x = Сократ" съждението е вярно
- Едно атомарно съждение, стойността на което се определя от един или повече участващи индивиди, се нарича предикат
- Второто съждение дава квантитативно твърдение за индивиди
- В терминологията на логиката може да се представи: „за всички x е в сила: ако Човек(x) е вярно съждение, тогава Смъртен(x) е също вярно съждение"

Предикатна логика

- Предикатната логика е разширение на съждителната логика със следното:
  - Многоместни предикати
  - Възможности за формулиране на квантифицирани съждения
- Нарича се предикатна логика от първа степен

1	Човек(Сократ)
2	$\forall$x (Човек(x) $\rightarrow$ Смъртен(x))
3	$\models$ Смъртен(Сократ)

Предикатна логика

- ***Логически квантори:*** За да изразим съждение като предходното, въвеждаме **квантовите променливи.**
- Ясно е, че едно твърдение със свободни променливи не е нито вярно, нито невярно, докато свободните променливи не получат стойности.
- ***Например:*** *"Ако $x \neq 0$, то $x.y = 1$".*

Ако x=2; y=1/2, твърдението е вярно.

Ако x=2;y=3, твърдението не е вярно.

- **Заб.** Когато правим тези замествания, винаги имаме предвид конкретна област на стойностите. Например: x,y – реални числа; а не x-марсианец, а y – Мики Маус.

Предикатна логика

- ***<u>Квантор за съществуване</u>***

  Как да изразим в математиката съществуването на нещо?

- Нека P е твърдение и нека съществуването на x означим с $\exists x$. Тогава

  $\exists x$:P е твърдението:

  ”Съществува една x, такава че P”. Променливата x е квантова променлива.

Предикатна логика

- Твърдението $\exists x{:}P$ е вярно, ако P е вярно за поне една стойност на x, избрана от нейната област (домейн).
- Символът $\exists$ е квантор за съществуване.

- ***<u>Пример</u>:*** *Нека P : $x^2 + 3x + 2 = 0$, x-свободна променлива.*

Верността на P зависи от стойността на x:

- *ако x=2, то P е F;*
- *ако x=-2, то P е T.*

Областта на стойностите на x е Z.

<u>Като запис: $\exists$ x : $x^2 + 3x + 2 = 0$.</u>

Предикатна логика

- ***Универсален квантор.*** -В много математически твърдения като:

“ За всяко $x \neq 0$, съществува у: х.у=1” се твърди, че за всяка стойност на свободната променлива х твърдението Р е в сила.

- ***Дефиниция:*** Нека Р е твърдение със свободна променлива х. Тогава $\forall x$:P е твърдение, което се чете: “За всяко х – Р”

Предикатна логика

- Променливата x е свързана във$\forall$ x:P, като $\forall$ x:P е вярно, ако P е вярно за всяка стойност на x от нейната област.
- "$\forall$" **е универсален квантор.**
- Вече можем да запишем и пълното твърдение:

$\forall$ x:(x $\neq$0 $\rightarrow$ $\exists$y:xy=1)

или $\forall$ x $\neq$0, $\exists$y:xy=1

Предикатна логика

Определяне верностните стойности на квантифицираните твърдения:

1) За да докажем, че $\forall$ x:P е вярно, трябва да го докажем за всички стойности на x.
2) За да докажем, че $\forall$ x:P е грешно, е достатъчно да докажем, че поне за една стойност на x е невярно.
3) За да докажем, че $\exists$x:P е вярно, е достатъчно да намерим само един случай за x, в който твърдението да е вярно.
4) За да докажем, че $\exists$x:P е грешно, е достатъчно да докажем, че не $\exists$x, така, че P да е вярно или, че P е грешно за $\forall$ x.

Предикатна логика

- **<u>Проблем</u>:** Ако е дадено едно квантифицирано съждение, може ли и неговото отрицание да запишем като квантифицирано съждение?

<u>Отговор:</u> Да!

<u>ТЕОРЕМА 1:</u> Нека P е твърдение. Тогава

$\neg(\forall x{:}P) \Leftrightarrow \exists x{:} \neg P$;

$\neg(\exists x{:}P) \Leftrightarrow \forall x{:} \neg P$;

Използвана литература в курса

- D. W. Hoffmann, Theoretische Informatik, Hansen Verlag, 2009
- H. P. Gumm, M. Sommer, Einfuehrung in die Informatik, Oldenbourg Wissenschaftsverlag, 2004
- J. W. Grossman, Discrete Mathematics, Macmillan Pub. Co., 1990
- К. Манев, Увод в дискретната математика, КЛМН, 2005
- Й. Денев, Р. Павлов, Я. Демирович. *Дискретна математика*. Наука и изкуство, София, 1984.

Използвана литература в курса

- Д. Байнов, С. Костадинов, Р. Павлов, Л. Луканова. *Ръководство за решаване на задачи по дискретна математика.* Университетско издателство “Паисий Хилендарски”, Пловдив, 1990.
- В.А. Успенский, *Машина Поста*, Москва, Наука, 1988, ISBN 5-02-013735-9.
- L. Lovasz, J. Pelikan, K. Vesztergombi, *Discrete Mathematics – Elementary and Beyond*, Springer Verlag, New York, 2003, ISBN 0-387-95584-4.

Използвана литература в курса

- E. Bender, S. Williamson, *A Short Course in Discrete Mathematics*, Dover, 2006, ISBN 0-486-43946-1.
- P. Linz, An *Introduction to Formal Languages and Automata,* Jones and Bartlett Publishers, 6-th edition, Jones & Bartlett Publishers, ISBN-13: 9781284077247, 2016
- Kenneth H. Rosen, Kamala Krithivasan, Discrete mathematics and its application, McGraw-Hill Companies, 7-th edition, ISBN 978-0-07-338309-5, 2012

Използвана литература в курса

- Owen D. Byer, Deirdre L. Smeltzer, Kenneth L. Wantz, Journey into Discrete Mathematics, AMS, MAA Press, Providence Rhode Island, ISBN 9781470446963, 2018
- Christopher Rhoades, Introductory Discrete Mathematics, Willford Press, ISBN 1682854922, 9781682854921, 2018
- David Liben-Nowell, Discrete Mathematics for Computer Science, Wiley, 2017, ISBN 1119397197, 9781119397199, 2017.
- ***http://www.jflap.org/*** - софтуерна среда