

*История информатики*

# АЛГОРИТМЫ

**М.А. Ройтберг**

**Часть 1**

**Понемногу обо всем**

**ФИВТ МФТИ**

*21 апреля 2017*

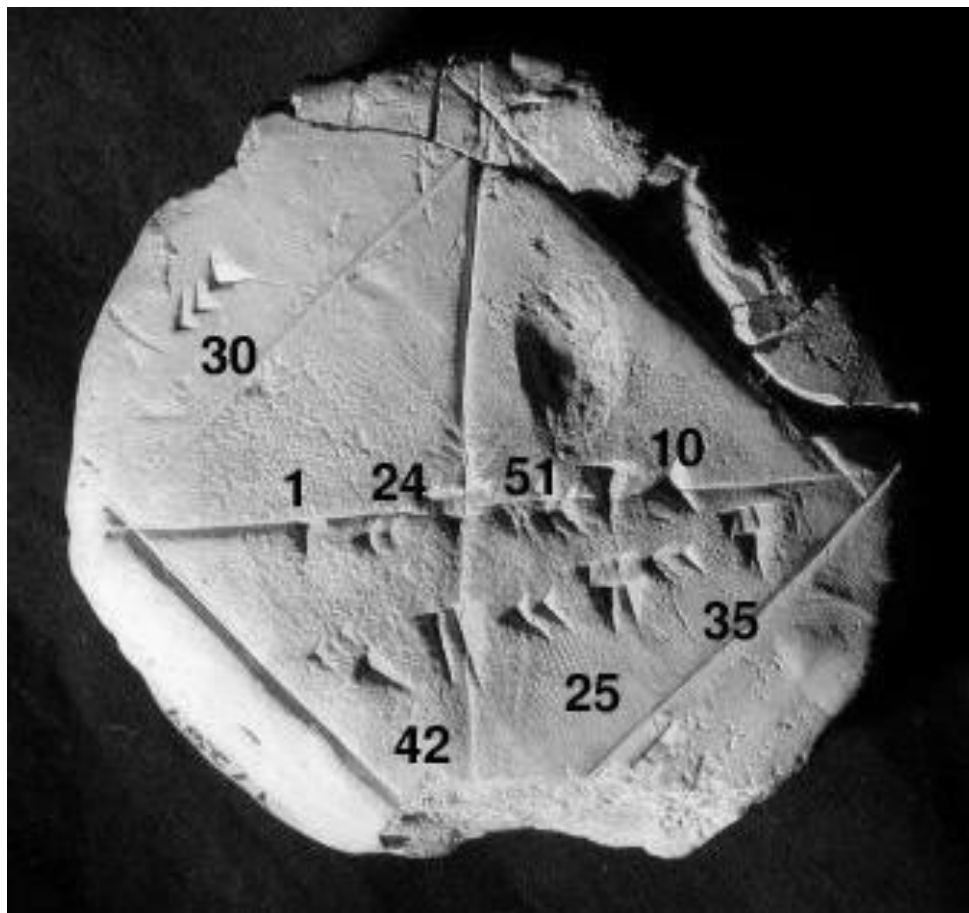


## Древний мир

- Алгоритмы – это алгоритмы операций с целыми числами (совсем древний мир).
  - "Натуральные числа создал Господь Бог. Все остальное - дело рук человеческих".  
( Леопольд Кронекер (1823-1891), немецкий математик «символ Кронекера»)
- Позже
  - алгоритмы работы с обыкновенными дробями;
  - алгоритмы вычисления приближений иррациональных чисел (корень из 2, число пи)

Вавилонская табличка (около 1800–1600 г. до н. э.)  
с вычислением

$$\sqrt{2} \approx 1 + 24/60 + 51/60^2 + 10/60^3 = 1.41421296\dots$$



## Древняя Греция (VI век до н. э. - V век н. э.)



Рафаэль Санти. Пифагор  
(деталь Афинской школы,  
1509-1511, Ватикан, Рим)

**Пифагор Самосский** (лат.*Pythagoras*, др.-греч. Πυθαγόρας ὁ Σάμιος; 570 —490 гг. до н. э.) — древнегреческий философ, математик и мистик, создатель религиозно-философской школы пифагорейцев

*«элементы чисел являются элементами всех вещей... и что весь мир в целом является гармонией и числом»*

Научились оперировать с дробями, находить общие делители и кратные

### **Кризис: открытие иррациональности корня из двух**





Рафаэль Санти. Пифагор  
(деталь Афинской школы,  
1509-1511, Ватикан, Рим)

## Древняя Греция (VI век до н. э. - V век н. э.)

**Пифагор Самосский** (лат. *Pythagoras*, др.-греч. Πυθαγόρας ὁ Σάμιος; 570 — 490 гг. до н. э.) — древнегреческий философ, математик и мистик, создатель религиозно-философской школы пифагорейцев

*«элементы чисел являются элементами всех вещей... и что весь мир в целом является гармонией и числом»*

Научились оперировать с дробями, находить общие делители и кратные

**Кризис: открытие иррациональности  
корня из двух**

**Следствие: недоверие к числам.  
Геометрическая алгебра.**



# Древняя Греция (VI век до н. э. - V век н. э.)

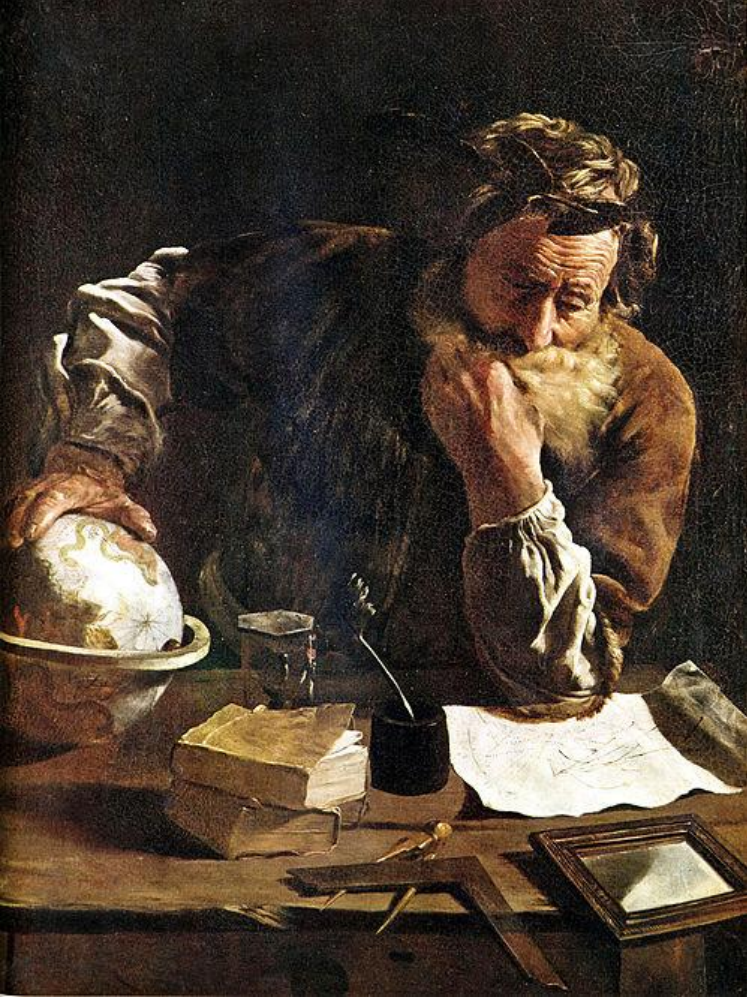
**Алгоритмы: геометрические построения**

**Три великие задачи древности:**

- удвоение куба;
- трисекция угла;
- квадратура круга

*Элементарные операции:* циркуль, лтнейка

## Древняя Греция (VI век до н. э. - V век н. э.)



«Архимед»  
(Доменико Фетти, 1620)

- **Архимед** (Ἀρχιμήδης; 287 до н. э. — 212 до н. э.) — древнегреческий математик, физик и инженер из Сиракуз. Сделал множество открытий в геометрии. Заложил основы механики, гидростатики, был автором ряда важных изобретений.
- «Дайте мне точку опоры и я переверну мир!»
- «Эврика!»
- «Не трогай мои чертежи!»



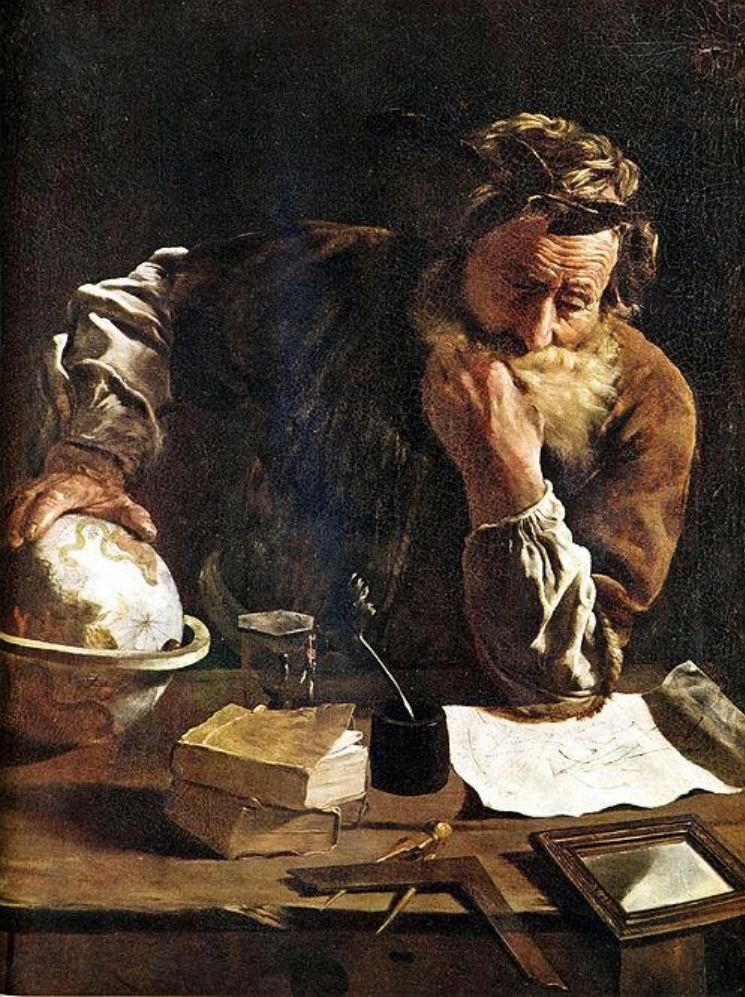
## Древняя Греция (VI век до н. э. - V век н. э.)

**Архимед** (Ἀρχιμήδης; 287 до н. э. — 212 до н. э.) — математик, физик и инженер

Суммирование бесконечного ряда

Площадь параболы

Позиционная система счисления



**«Архимед»  
(Доменико Фетти, 1620)**





# Древняя Греция (VI век до н. э. - V век н. э.)

~ 300 до н.э.

## Алгоритм Евклида

**Евкли́д** или **Эвкли́д** (др.-греч. Εὐκλείδης, от «добрая слава», время расцвета — около 300 года до н. э.) — древнегреческий математик, автор первого из дошедших до нас теоретических трактатов по математике.

~ 200 до н.э.

## Решето Эратосфена

**Эратосфэ́н Кире́нский** (Ἐρατοσθένης ὁ Κιρηναῖος; 276 год до н. э.—194 год до н. э.) — греческий математик, астроном, географ, филолог и поэт. С 235 г. до н. э. — глава Александрийской библиотеки. Первый известный учёный, вычисливший размеры Земли.



## Арифметика

Позиционная система счисления и алгоритмы выполнения арифметических операций в ней – одни из важнейших алгоритмических открытий

**Арифметика. Роспись Пинтуриккьо. Апартаменты Борджиа. 1492—1495. Рим, Ватиканские дворцы**





# Арабский мир (IX-XIV века)

- c. 820 - **Al-Khwarizmi** described algorithms for solving linear equations and quadratic equations in his Algebra;
- 825 - **Al-Khwarizmi** described the algorithm, algorithms for using the Hindu-Arabic numerals, in his treatise On the Calculation with Hindu Numerals, which was translated into Latin as Algoritmi de numero Indorum, where "Algoritmi", the translator's rendition of the author's name gave rise to the word algorithm (Latin algorithmus) with a meaning "calculation method"
- c. 850 - Cryptanalysis and frequency analysis algorithms developed by **Al-Kindi (Alkindus)** in A Manuscript on Deciphering Cryptographic Messages, which contains algorithms on breaking encryptions and ciphers.
- c. 1025 - **Ibn al-Haytham (Alhazen)**, was the first mathematician to derive the formula for the sum of the fourth powers, and in turn, he develops an algorithm for determining the general formula for the sum of any integral powers, which was fundamental to the development of integral calculus[c. 1400 - **Ahmad al-Qalqashandi** gives a list of ciphers in his Subh al-a'sha which include both substitution and transposition; he also gives an example of cryptanalysis, including the use of tables of letter frequencies and sets of letters which can not occur together in one word





## А тем временем...

- 263 – Китай: решение систем линейных уравнений методом Гаусса (Liu Hui)
- 628 – Индия: решение диофантовых уравнений, в т. ч. уравнения Пелля (Brahmagupta)
- ... и многое другое

## ВОЗРОЖДЕНИЕ XIV – XVI века



**Галилео Галилей**  
(с. 1630), Петер Пауль Рубенс

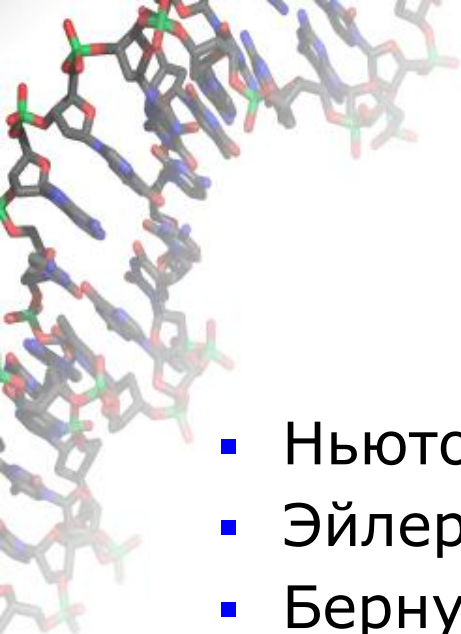
- **Галилэ́ Галилэ́й** (итал.*Galileo Galilei*; 1564, Пиза 1642, Арчетри) физик, механик, астроном, философ и математик. Оказал огромное влияние на науку. Он первым использовал телескоп для наблюдения небесных тел и сделал ряд выдающихся астрономических открытий. Галилей — основатель экспериментальной физики; заложил фундамент классической механики.



## Уравнения 3-й и 4-й степени

- **Сципион дель Ферро** (итал. *Scipione del Ferro*, 6 февраля 1465, Болонья, северная Италия — 5 ноября 1526, там же) — итальянский математик, открывший общий метод решения неполного кубического уравнения.
- **Джерола́мо** (Джироламо, Иероним) **Карда́но** (лат. *Hieronymus Cardanus*, итал. *Girolamo Cardano*, *Gerolamo Cardano*; 24 сентября 1501, Павия — 21 сентября 1576, Рим) — итальянский математик, инженер, философ, медик и астролог. В его честь названы открытые Сципионом дель Ферро формулы решения кубического уравнения (Кардано был их первым публикатором), карданов подвес и карданный вал.
- **Лодовико** (Луиджи) **Феррари** (итал. *Lodovico Ferrari*; 2 февраля 1522 года, Болонья — 5 октября 1565 года) — итальянский математик, нашедший общее решение уравнения четвёртой степени.

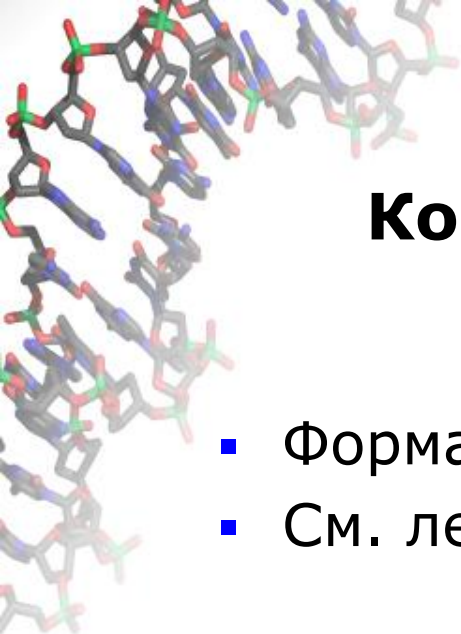




## XVII – XIX века

- Ньютон,
- Эйлер,
- Бернулли
- Гаусс ...
- ... и многие другие

### ■ **Вычислительные алгоритмы**



## **Конец XIX – 1-я половина XX века**

- Формализация понятия алгоритма
- См. лекции А.Л. Семенова

# 1940-е

## Модели вычислений: архитектура компьютеров

- **1944-1945** – компьютер с хранимой программой (проект EDVAC)  
EDVAC (Electronic Discrete Variable Automatic Computer) — одна из первых электронных вычислительных машин. В отличие от своего предшественника ЭНИАКа, это был компьютер на двоичной, а не десятичной основе. Как и ЭНИАК, EDVAC был разработан в Институте Мура Пенсильванского Университета для Лаборатории баллистических исследований Армии США командой инженеров и учёных во главе с Джоном Преспером Экертом и Джоном Уильямом Мокли при активной помощи математиков Джона фон Неймана и Германа Голдстайна.  
1944 – проект Эккерта  
1945 – отчет фон Неймана





# 1940-е

## Модели вычислений: модель нейрона

- 1943 – модель нейрона МакКаллока-Питтса

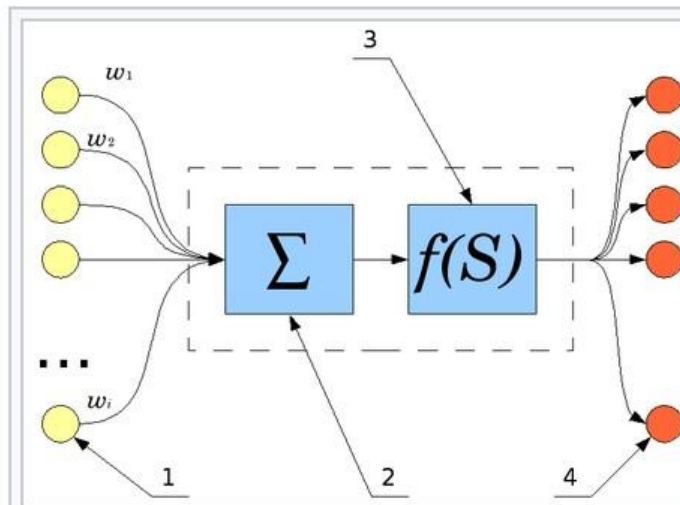


Схема искусственного нейрона

1. Нейроны, выходные сигналы которых поступают на вход данному
2. Сумматор входных сигналов
3. Вычислитель передаточной функции
4. Нейроны, на входы которых подаётся выходной сигнал данного
5.  $w_i$  — веса входных сигналов

# 1952-1955

## Модели вычислений: конечные автоматы

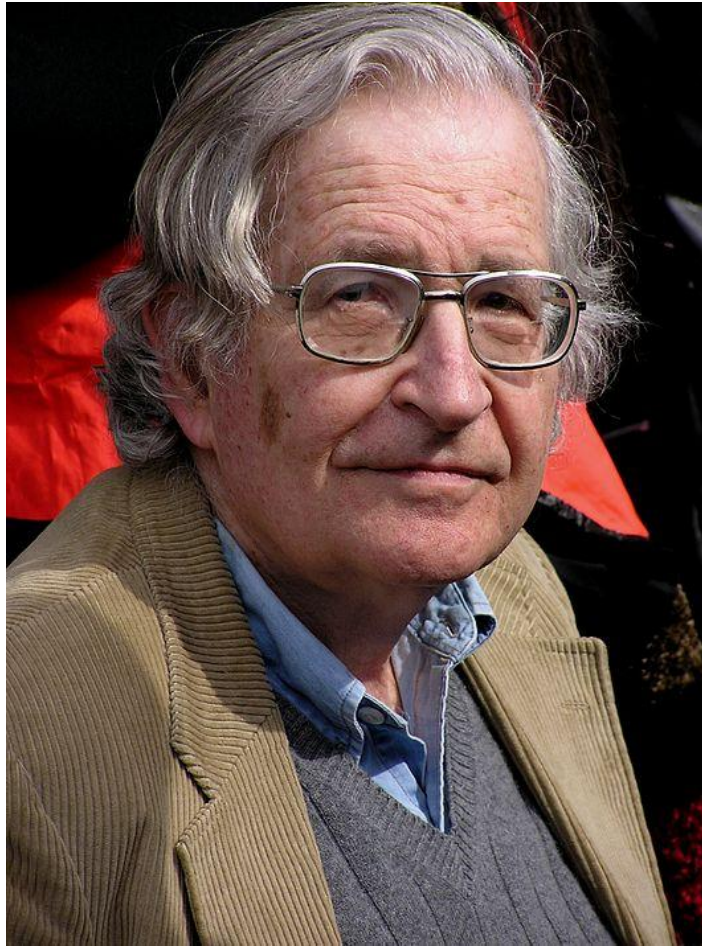


CONTENTS	
Preface	v
FINITE AUTOMATA	
Representation of Events in Nerve Nets and Finite Automata By S. C. Kleene	1
Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components By J. von Neumann	43
Some Uneconomical Robots By James T. Culbertson	99
Some Universal Elements for Finite Automata By M. L. Minsky	117
Gedanken-Experiments on Sequential Machines By Edward F. Moore	129
TURING MACHINES	
A Universal Turing Machine with Two Internal States By Claude E. Shannon	157
A Note on Universal Turing Machines By M. D. Davis	167
The Inversion of Functions Defined by Turing Machines By John McCarthy	177
Computability by Probabilistic Machines By E. de Leeuw, E. F. Moore, C. E. Shannon, and N. Shapiro	183
SYNTHESIS OF AUTOMATA	
Design for an Intelligence-Amplifier By W. Ross Ashby	215
The Epistemological Problem for Automata By D. M. MacKay	235
Conditional Probability Machines and Conditioned Reflexes By Albert M. Uttley	251
Temporal and Spatial Patterns in a Conditional Probability Machine By Albert M. Uttley	277

**1956 – Automata studies. Ed. By C. Shannon & J. McCarthy.**

**1957**

# **Модели вычислений: порождающие грамматики Хомского**



Аврам Ноам Хомский (англ. Avram Noam Chomsky ['noʊm 'tʃɒmski]; 7 декабря 1928, Филадельфия, штат Пенсильвания, США) — американский лингвист, <sup>20</sup> политический публицист, философ и теоретик.

# 1956-1960

## Модели вычислений: искусственный интеллект



CONTENTS	
Preface	v
FINITE AUTOMATA	
Representation of Events in Nerve Nets and Finite Automata By S. C. Kleene	1
Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components By J. von Neumann	43
Some Uneconomical Robots By James T. Culbertson	99
Some Universal Elements for Finite Automata By M. L. Minsky	117
Gedanken-Experiments on Sequential Machines By Edward F. Moore	129
TURING MACHINES	
A Universal Turing Machine with Two Internal States By Claude E. Shannon	157
A Note on Universal Turing Machines By M. D. Davis	167
The Inversion of Functions Defined by Turing Machines By John McCarthy	177
Computability by Probabilistic Machines By E. de Leeuw, E. F. Moore, C. E. Shannon, and N. Shapiro	183
SYNTHESIS OF AUTOMATA	
Design for an Intelligence-Amplifier By W. Ross Ashby	215
The Epistemological Problem for Automata By D. M. MacKay	235
Conditional Probability Machines and Conditioned Reflexes By Albert M. Uttley	251
Temporal and Spatial Patterns in a Conditional Probability Machine By Albert M. Uttley	277

**1956 – Automata studies. Ed. By C. Shannon & J. McCarthy.**





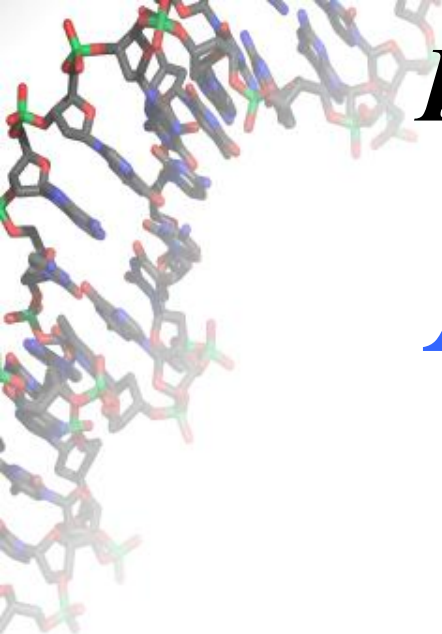
# 1950-е

## Модели вычислений: искусственный интеллект

- 1955 – появление термина AI (John McCarthy)
- 1956 – Дартмутская конференция (организаторы: McCarthy, Marvin Minsky, Nathaniel Rochester and Claude Shannon)
- 1955 - 1956 – Logic Theorist (Allen Newell, Herbert A. Simon)
- 1957 – 1959 - General Problem Solver )Herbert A. Simon, J.C. Shaw, and Allen Newell)
- !!! C. Barnard and H. A. Simon. (1947). Administrative behavior. A study of decision-making processes in administrative organization. Macmillan, New York. => Нобелевская премия 1978 г.

# 1950+ - алгоритмы в современном понимании

- 1949 - 1952 - dynamic programming (R. Bellman)
- 1952 - Huffman codes (David A. Huffman)
- 1953 - Simulated annealing introduced by Nicholas Metropolis
- 1954 - Radix sort computer algorithm developed by Harold H. Seward
- 1956 - Kruskal's algorithm developed by Joseph Kruskal
- 1957 - Prim's algorithm developed by Robert Prim
- 1957 - Bellman–Ford algorithm developed
- 1959 - Dijkstra's algorithm developed by Edsger Dijkstra
- 1959 - Shell sort developed by Donald L. Shell
- 1960 - Karatsuba multiplication
- 1962 - AVL trees
- 1962 - Quicksort developed by C. A. R. Hoare
- 1962 - Ford–Fulkerson algorithm
- 1964 - Heapsort developed by J. W. J. Williams
- 1964 - multigrid methods first proposed by R. P. Fedorenko
- 1965 - Cooley–Tukey algorithm
- 1965 - Levenshtein distance developed by Vladimir Levenshtein
- 1965 - Cocke–Younger–Kasami (CYK)
- 1967 - Viterbi algorithm proposed by Andrew Viterbi

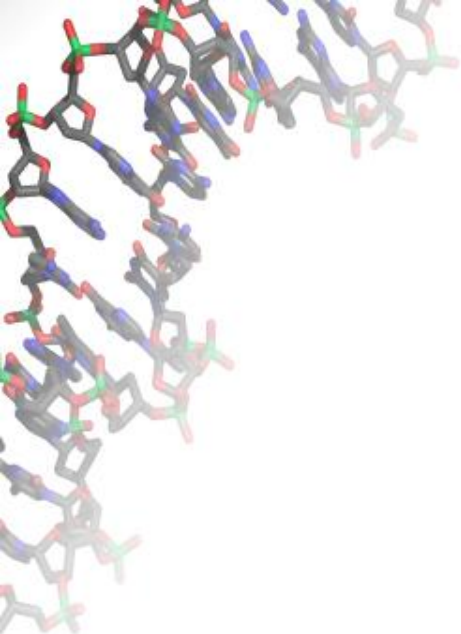


*История информатики*

# Алгоритмы

**Часть 2**

**Динамическое  
программирование - 2016**



# **0. Повторение. И немного нового**



# Эпиграф.

*Найти сумму 1 000 000 слагаемых за 5000 операций*

$S =$

$$= a_1 \cdot b_1 + a_1 \cdot b_2 + \dots + a_1 \cdot$$

$$b_{1000} +$$

$$+ a_2 \cdot b_1 + a_2 \cdot b_2 + \dots + a_2 \cdot$$

$$b_{1000} +$$

+

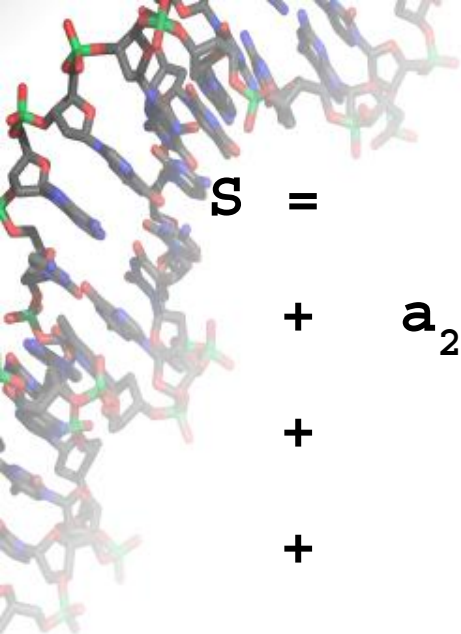
...

+

$$+ a_{1000} \cdot b_1 + a_{1000} \cdot b_2 + \dots + a_{1000} \cdot$$

$$b_{1000}$$

# Решение


$$\begin{aligned} S &= a_1 \cdot (b_1 + b_2 + \dots + b_{1000}) + \\ &+ a_2 \cdot (b_1 + b_2 + \dots + b_{1000}) + \\ &+ \dots \\ &+ a_{1000} \cdot (b_1 + b_2 + \dots + b_{1000}) = \\ &= (a_1 + a_2 + \dots + a_{1000}) \cdot (b_1 + b_2 + \\ &\dots + b_{1000}) \end{aligned}$$

\*\*\* Алгоритм \*\*\*

$$\mathbf{A} = a_1 + a_2 + \dots + a_{1000} \quad // \quad 999$$

операций

$$\mathbf{B} = b_1 + b_2 + \dots + b_{1000} \quad // \quad 999$$

# Ориентированный ациклический граф с весами на ребрах

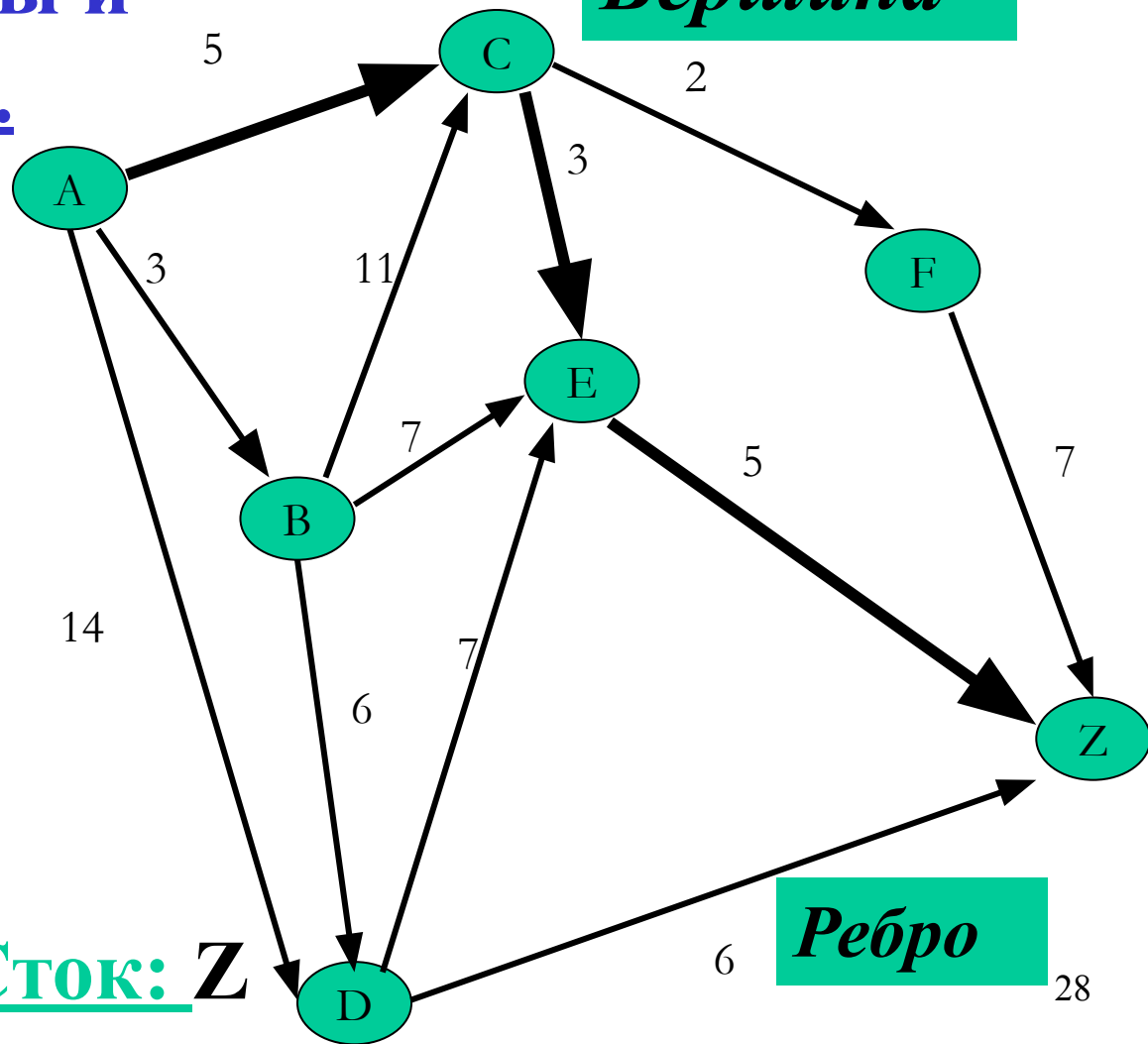
Ребра направлены и снабжены весами.

Путь: **ABCE**  
 $W(ABCE) =$   
 $= 3 + 11 + 3 = 17$

Нет циклов

Источник: A;

Сток: Z



## Пути (примеры):

$BEZ = \{(BE), (EZ)\}$  (длина 2);

$$\text{вес } W(BEZ) = 7 + 5 = 12$$

*мультипликативный вес (м-вес)*

$$WM(BEZ) = 7 \cdot 5 = 35$$

$BCEZ = \{(BC), (CE), (EZ)\}$  (длина 3);

$$W(BCEZ) = 11 + 3 + 5 = 19$$

$$WM(BCEZ) = 11 \cdot 3 \cdot 5 = 165$$

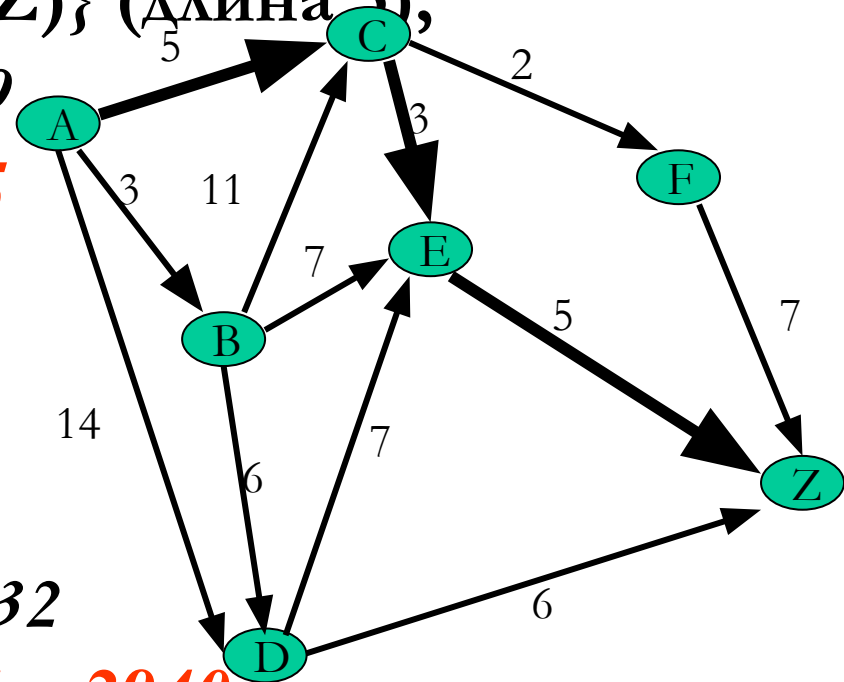
Полный путь (длина 4); :

$ADBEZ =$

$= \{(AD), (DB), (BE), (EZ)\}$

$$W(ADBEZ) = 14 + 6 + 7 + 5 = 32$$

$$WM(ADBEZ) = 14 \cdot 6 \cdot 7 \cdot 5 = 2940$$





*Полные пути –*

*пути из источника в сток (примеры):*

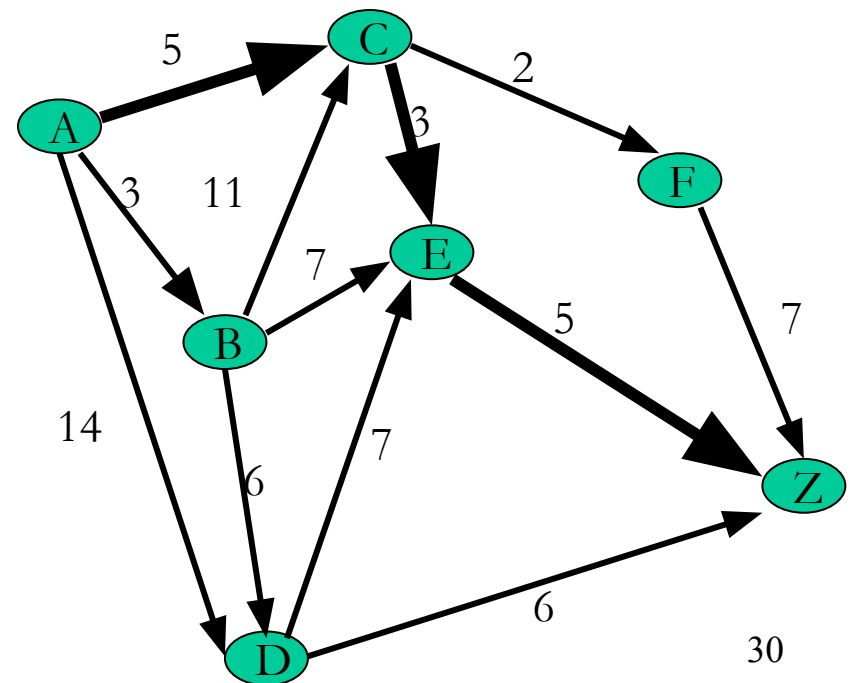
**ADEZ:** длина = 3; вес  $W(ADEZ) = 14 + 7 + 5 = 26$ ;

*м-вес  $WM(ADEZ) = 14 \cdot 7 \cdot 5 = 490$*

**ABCFZ:** длина = 4;

$W(ABCFZ) = 3 + 7 + 2 + 7 = 19$

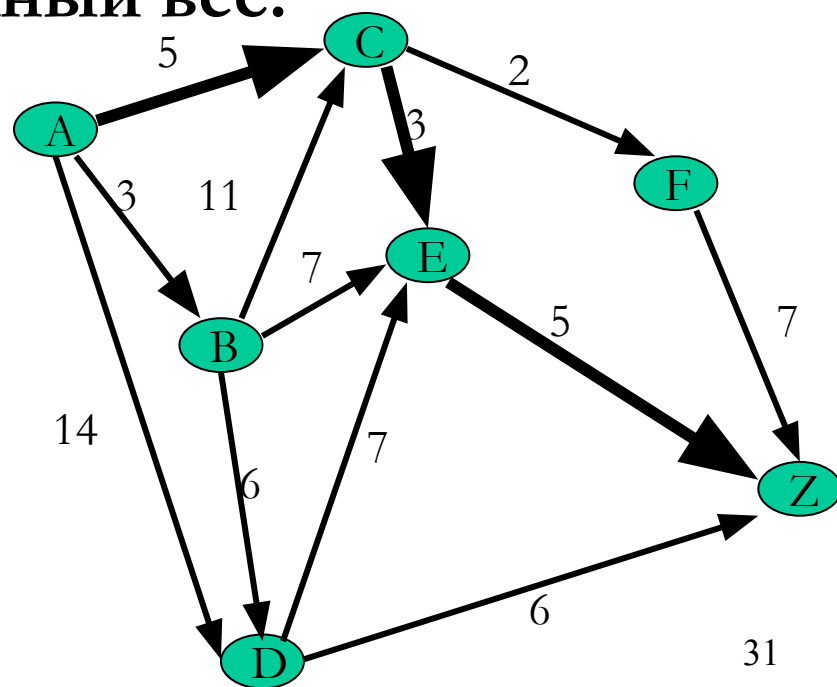
*$WM(ADEZ) = 3 \cdot 7 \cdot 2 \cdot 7 = 294$*



ДАНО: Ориентированный ациклический граф с весами на ребрах

$$G = \langle V, E, W; A, Z \rangle$$

ЗАДАЧА 1 Найти оптимальный полный путь, т.е. полный путь, имеющий минимальный (максимальный) возможный вес.



# Метод динамического программирования (Алгоритм Беллмана, 1953)

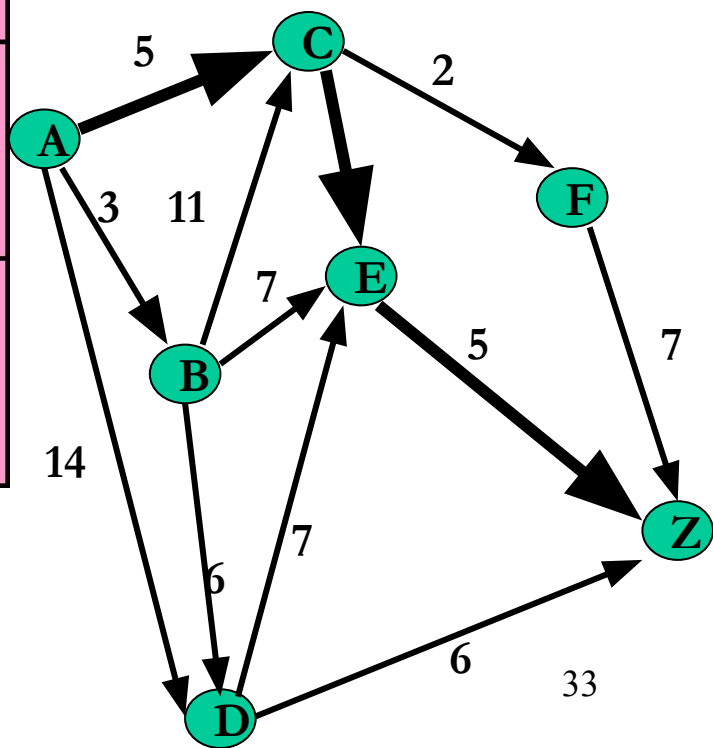
- Проход от стока к источнику:  
из  $W$  есть путь в  $V \Rightarrow$   
 $\Rightarrow W$  обрабатывается позже, чем  $V$ .
- Рекуррентное уравнение

$$\mathbf{BestW(A) = \min\{$$
$$\mathbf{W(AB) + BestW(B),}$$
$$\mathbf{W(AC) + BestW(C),}$$
$$\mathbf{W(AD) + BestW(D)}$$
$$\mathbf{\}}$$



Ранг	Вершина	Исх. ребра	Вес ребра	Вес след. верш.	Лучш вес для ребра	Лучш вес для верш	Куда идти
0	Z	xxx	0	0	0	0	xxx
1	F	Z	7	0	7	7	Z
1	E	Z	5	0	5	5	Z
2	D	E	7	5	12	6	Z
		Z	6	0	6		
2	C	E	3	5	8	8	E
		F	2	7	9		
3	B	C	11	8	19	12	D, E
		D	6	6	12		
		E	7	5	12		
4	A	B	3	12	15	13	C
		C	5	8	13		
		D	14	6	20		

$$\begin{aligned}
 \text{BestW}(B) &= \\
 &= \min\{ \\
 &\quad W(BC) + \text{BestW}(C), \\
 &\quad W(BD) + \text{BestW}(D), \\
 &\quad W(BE) + \text{BestW}(E), \\
 &\quad \}
 \end{aligned}$$



**Best Weight: 13**  
**Best Path: ACEZ**



## 1.2. Мультипликативные веса путей

$BEZ = \{(BE), (EZ)\}$  (длина 2);

вес  $W(BEZ) = 7 + 5 = 12$

*мультипликативный вес (м-вес)*

$$WM(BEZ) = 7 \cdot 5 = 35$$

$BCEZ = \{(BC), (CE), (EZ)\}$  (длина 3);

$W(BCEZ) = 11 + 3 + 5 = 19$

$$WM(BCEZ) = 11 \cdot 3 \cdot 5 = 165$$

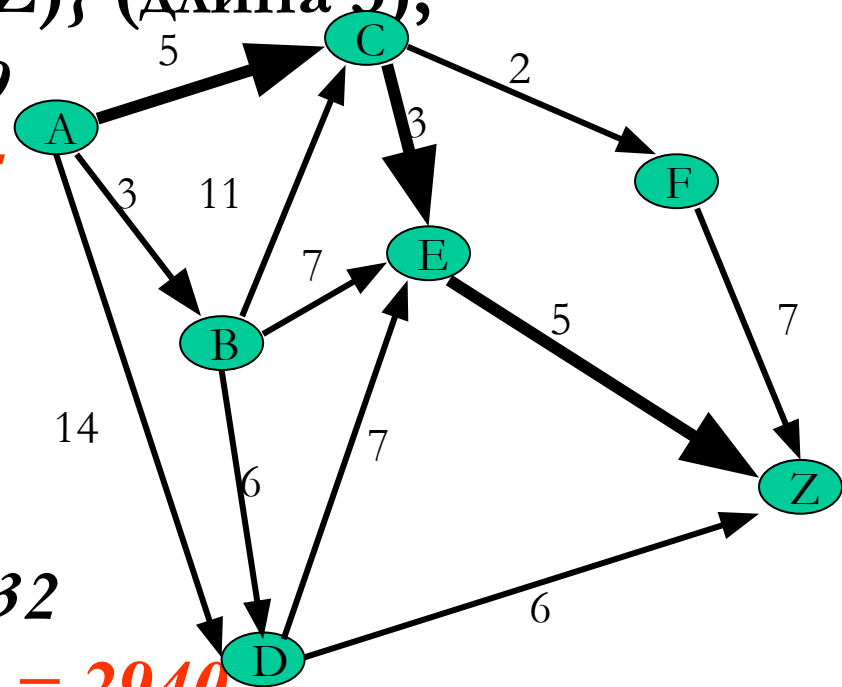
Полный путь (длина 4); :

$ADBEZ =$

$= \{(AD), (DB), (BE), (EZ)\}$

$W(ADBEZ) = 14 + 6 + 7 + 5 = 32$

$$WM(ADBEZ) = 14 \cdot 6 \cdot 7 \cdot 5 = 2940$$



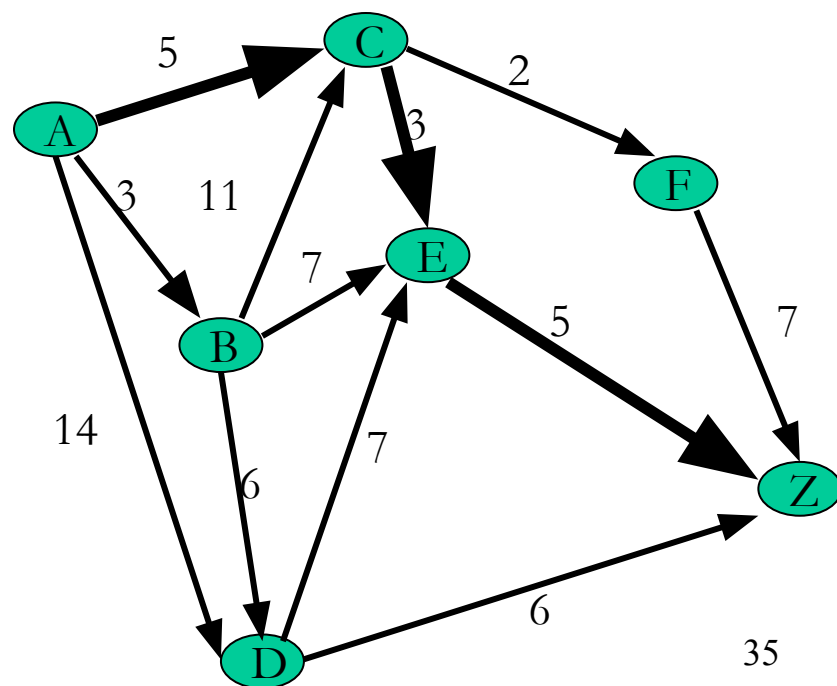
ДАНО: Ориентированный ациклический граф с весами на ребрах

$$G = \langle V, E, W; A, Z \rangle$$

ЗАДАЧА 2 («задача Больцмана») Найти сумму мультипликативных весов всех полных путей.

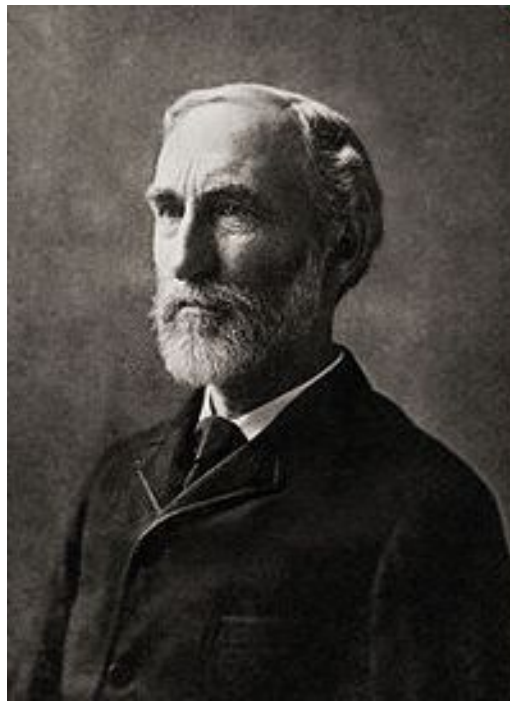


**Людвиг Больцман** (нем. Ludwig Eduard Boltzmann, 1844 - 1906), основатель статистической механики и молекулярно-кинетической теории





**Людвиг Больцман**  
(Ludwig Eduard Boltzmann, 1844 – 1906; Австро-Венгрия, Италия), основатель статистической механики и молекулярно-кинетической теории



■ **Джозайя Уиллард Гиббс** (*Josiah Willard Gibbs*; 1839 – 1903, США) — математик, физик и физикохимик, один из создателей статистической физики и математической теории термодинамики



**Эрнст Изинг** (Ernst Ising, 1900-1998, Германия-США) - физик, позже - педагог, автор модели Изинга (см. предсказание спиралей в белке и т. п.)

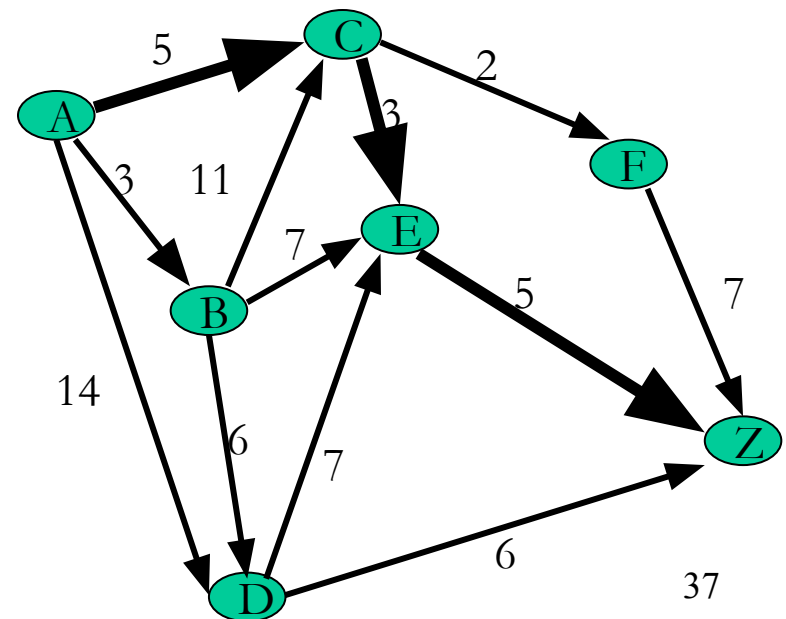
## *Интерпретации:*

*1. Вероятность прохода лабиринта:*

*Вершины – города; Ребра - дороги;*

*Вес ребра: вероятность перехода по ребру  
(сумма вероятностей выхода из вершины  
может быть меньше 1)*

*2. Статистическая  
физика – без  
комментариев*



Проход от стока к источнику:

из  $W$  есть путь в  $V \Rightarrow$

$\Rightarrow W$  обрабатывается позже, чем  $V$ .

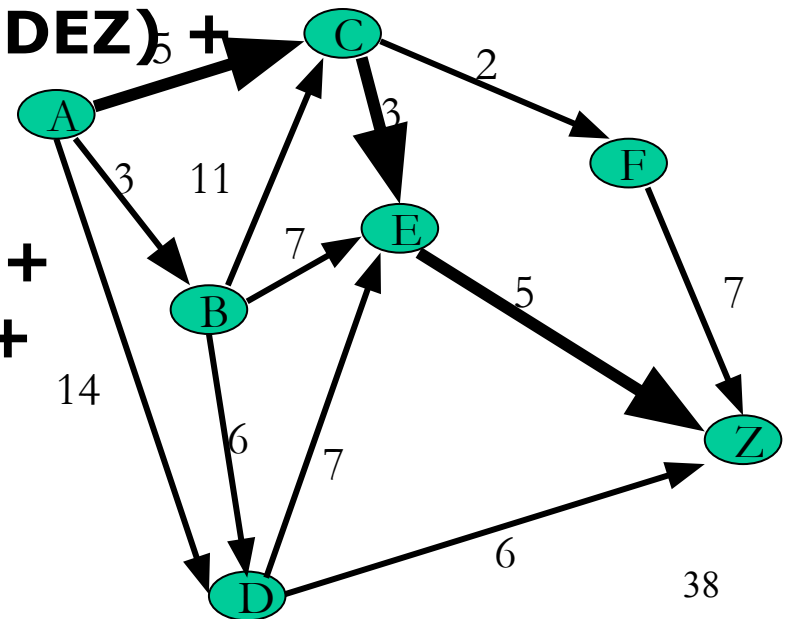
Пример: вершина  $B$ .

Пути из  $B$  в  $Z$ :  $BCEZ$ ,  $BCFZ$ ,  $BDZ$ ,  $BDEZ$ ,  $BEZ$

$$\begin{aligned} \text{Sum}(B) = & M(BCEZ) + M(BCFZ) + \\ & + M(BDZ) + M(BDEZ) + \\ & + M(BEZ) = \end{aligned}$$

$$\begin{aligned} = & W(BC) * M(CEZ) + W(BC) * M(CFZ) + \\ & + W(BD) * M(DZ) + W(BD) * M(DEZ) + \\ & + W(BE) * M(EZ) = \end{aligned}$$

$$\begin{aligned} = & W(BC) * (M(CEZ) + M(CFZ)) + \\ & + W(BD) * (M(DZ) + M(DEZ)) + \\ & + W(BE) * M(EZ) = \dots \end{aligned}$$





Проход от стока к источнику:

из  $W$  есть путь в  $V \Rightarrow$

$\Rightarrow W$  обрабатывается позже, чем  $V$ .

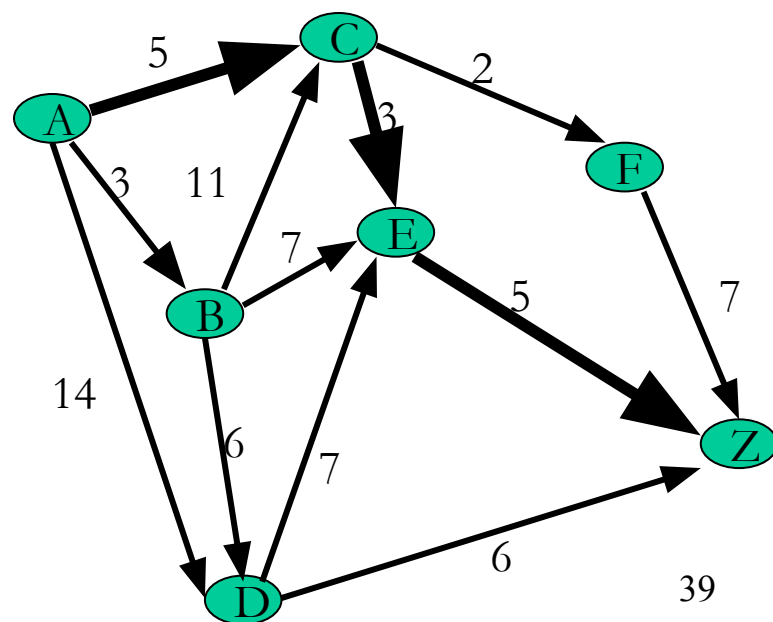
Пример: вершина  $B$ .

■ Пути из  $B$  в  $Z$ :  $BCEZ$ ,  $BCFZ$ ,  $BDZ$ ,  $BDEZ$ ,  $BEZ$

■ **Sum(B) = ...**

$$\begin{aligned} &= W(BC) * (M(CEZ) + M(CFZ)) + \\ &+ W(BD) * (M(DZ) + M(DEZ)) + \\ &+ W(BE) * M(EZ) = \end{aligned}$$

$$\begin{aligned} &= W(BC) * \text{Sum}(C) + \\ &+ W(BD) * \text{Sum}(D) + \\ &+ W(BE) * \text{Sum}(E) \end{aligned}$$



Проход от стока к источнику:

из  $W$  есть путь в  $V \Rightarrow$

$\Rightarrow W$  обрабатывается позже, чем  $V$ .

■ Пример: вершина  $B$ .

■ Пути из  $B$  в  $Z$ :  $BCEZ$ ,  $BCFZ$ ,  $BDZ$ ,  $BDEZ$ ,  $BEZ$

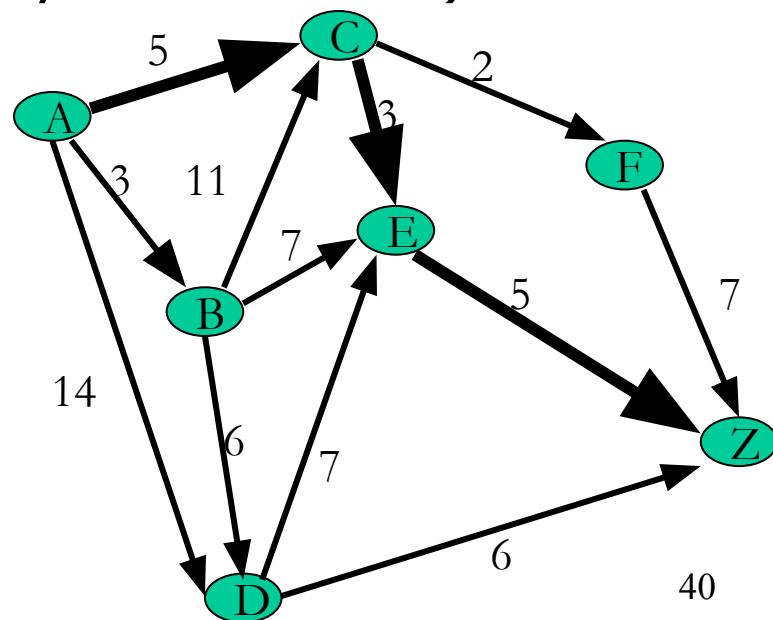
■  $\text{Sum}(B) = M(BCEZ) + M(BCFZ) +$   
 $+ M(BDZ) + M(BDEZ) +$   
 $+ M(BEZ) =$

■ Рекуррентное уравнение (сумма  $m$ -весов):

$\text{Sum}(A) =$

$W(AB) * \text{Sum}(B) +$   
 $+ W(AC) * \text{Sum}(C) +$   
 $+ W(AD) * \text{Sum}(D)$

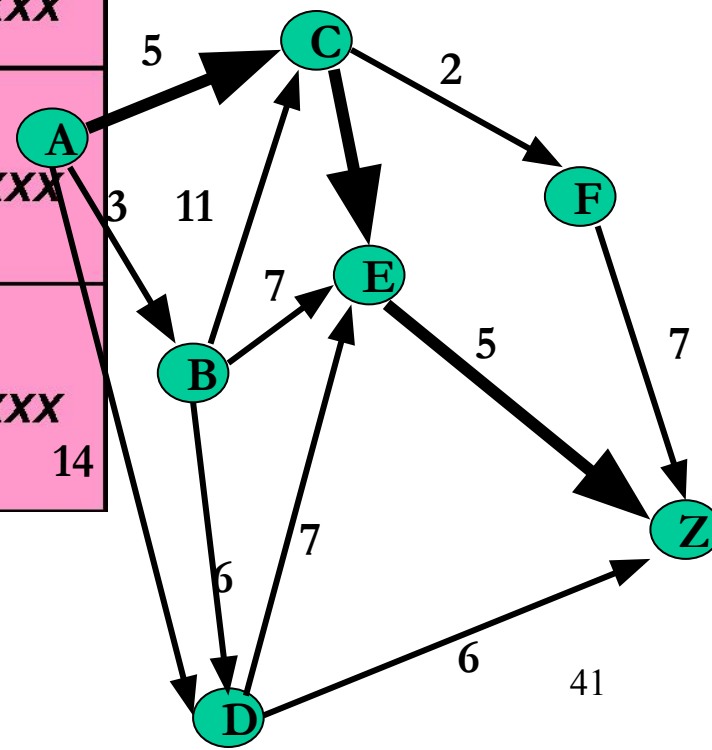
}



Ранг	Вершина	Исх. ребра	Вес ребра	Вес след. верш.	Сумм. вес для путей через ребро	Сумм. вес для верш	Куда идти
0	Z	xxx	xxx	xxx	xxx	1	
1	F	Z	7	1	7	7	
1	E	Z	5	1	5	5	
2	D	E	7	5	35	41	
		Z	6	1	6		
2	C	E	3	5	15	29	xxx
		F	2	7	14		
3	B	C	11	29	319	600	xxx
		D	6	41	246		
		E	7	5	35		
4	A	B	3	600	1800	2519	xxx
		C	5	29	145		
		D	14	41	574		

**Sum: 2519**

**BestFwW(B) =**  
**= summa{**  
**W(BC) \* BestFwW(C),**  
**W(BD) \* BestFwW(D),**  
**W(BE) \* BestFwW(E),**  
**}**



Проход от стока к источнику:

из  $W$  есть путь в  $V \Rightarrow$

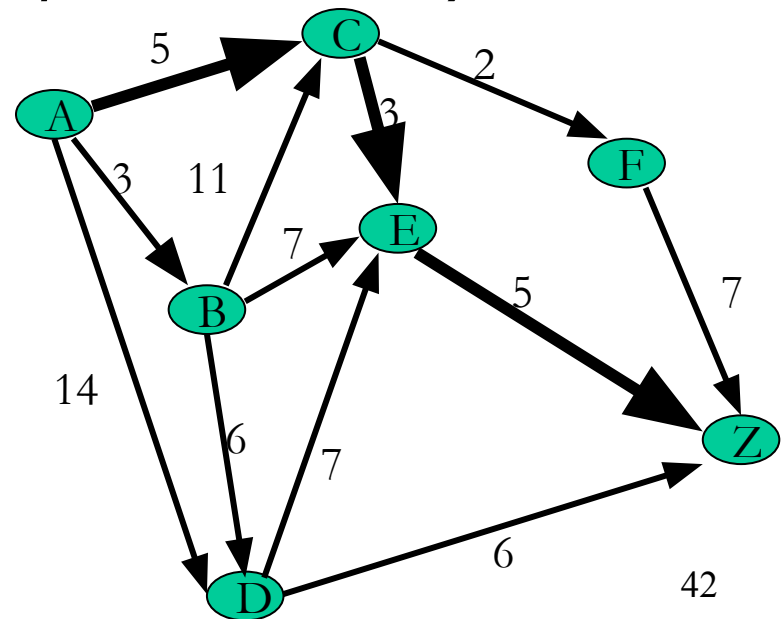
$\Rightarrow W$  обрабатывается позже, чем  $V$ .

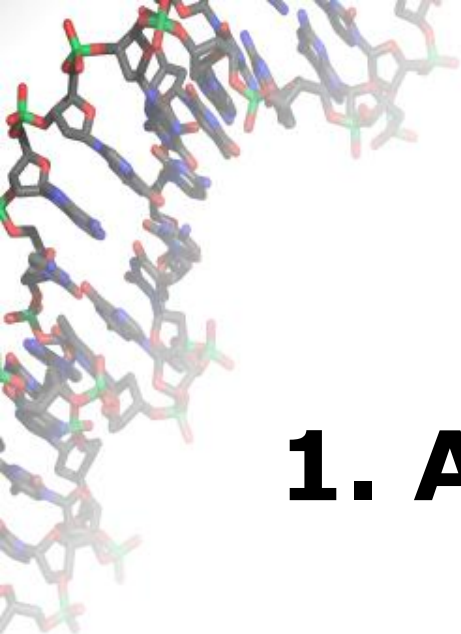
- Рекуррентное уравнение (минимальный путь)

$$\text{BestW}(A) = \min\{\begin{aligned} &W(AB) + \text{BestW}(B), \\ &W(AC) + \text{BestW}(C), \\ &W(AD) + \text{BestW}(D) \end{aligned}\}$$

- Рекуррентное уравнение (сумма м-весов):

$$\text{Sum}(A) = \begin{aligned} &W(AB) * \text{Sum}(B) + \\ &+ W(AC) * \text{Sum}(C) + \\ &+ W(AD) * \text{Sum}(D) \end{aligned}$$





# **1. Алгебраическая основа динамического программирования**



# Повторение: 1-й класс

 **Сочетательный закон (ассоциативность):**

**Сложение**

$$(a+b)+c = a+(b+c)$$

**Умножение**

$$(a*b)*c = a*(b*c)$$

**Переместительный закон (коммутативность):**

**Сложение**

$$a+b = b+a$$

**Умножение**

$$a*b = b*a$$

**Нейтральный элемент:**

**Сложение**

$$a+0 = 0+a = a$$

**Умножение**

$$a*1 = 1*a = a$$

**Обратные элементы (3-й класс ):**

**Сложение**

$$a+(-a) = 0$$

**Умножение**

$$a*(1/a) = 1a$$

■

# Повторение: 1-й класс



**Сочетательный закон (ассоциативность):**

**Сложение**

$$(a+b)+c = a+(b+c)$$

**Умножение**

$$(a*b)*c = a*(b*c)$$

**Переместительный закон (коммутативность):**

**Сложение**

$$a+b = b+a$$

**Умножение**

$$a*b = b*a$$

**Нейтральный элемент:**

**Сложение**

$$a+0 = 0+a = a$$

**Умножение**

$$a*1 = 1*a = a$$

▪ **Обратные элементы (3-й класс 😊) :**

**Сложение**

$$a+(-a) = 0$$

**Умножение**

$$a*(1/a) = 1a$$

▪ **РАСПРЕДЕЛИТЕЛЬНЫЙ ЗАКОН (ДИСТРИБУТИВНОСТЬ)**

**умножение относительно сложения**

$$(a+b)*c = a*c + b*c \quad a*(b+c) = a*b+a*c$$

# Что использовали?

**Сочетательный закон (ассоциативность):**

**Сложение**

$$(a+b)+c = a+(b+c)$$

**Умножение**

$$(a*b)*c = a*(b*c)$$

**Переместительный закон (коммутативность):**

**Сложение**

$$a+b = b+a$$

**Умножение**

$$a*b = b*a$$

**Нейтральный элемент:**

**Сложение**

$$a+0 = 0+a = a$$

**Умножение**

$$a*1 = 1*a = a$$

**Обратные элементы :**

**Сложение**

$$a+(-a) = 0$$

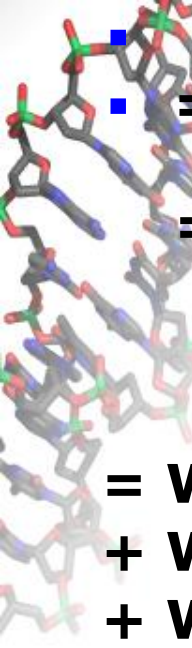
**Умножение**

$$a*(1/a) = 1$$

**РАСПРЕДЕЛИТЕЛЬНЫЙ ЗАКОН (ДИСТРИБУТИВНОСТЬ)**

**умножение относительно сложения**

$$(a+b)*c = a*c + b*c \quad a*(b+c) = a*b+a*c$$



$$\text{Sum}(B) =$$

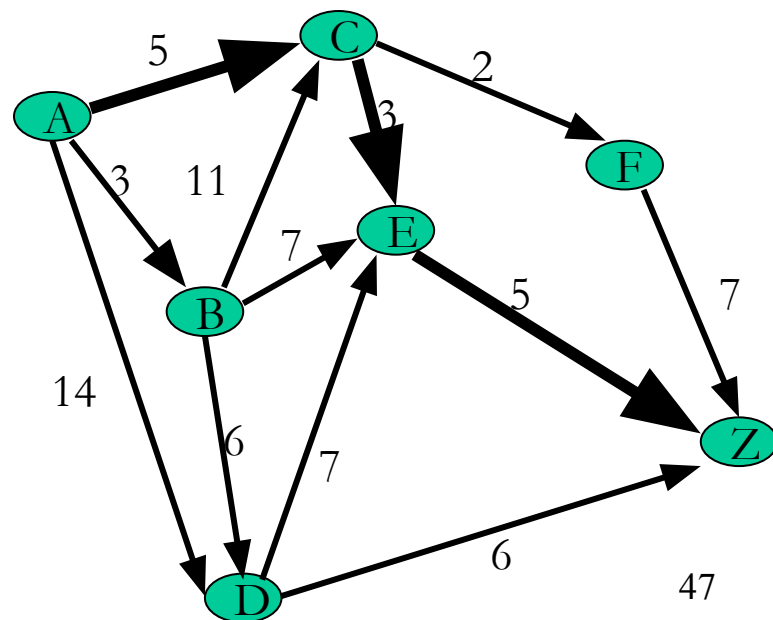
$$= M(\text{BCEZ}) + M(\text{BCFZ}) + M(\text{BDZ}) + M(\text{BDEZ}) + M(\text{BEZ}) =$$

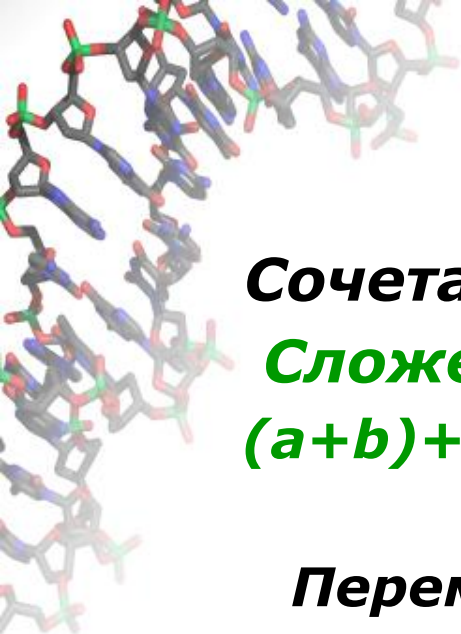
$$= W(\text{BC}) * M(\text{CEZ}) + W(\text{BC}) * M(\text{CFZ}) + \\ + W(\text{BD}) * M(\text{DZ}) + W(\text{BD}) * M(\text{DEZ}) + \\ + W(\text{BE}) * M(\text{EZ}) =$$

$$= W(\text{BC}) * (M(\text{CEZ}) + M(\text{CFZ})) + \\ + W(\text{BD}) * (M(\text{DZ}) + M(\text{DEZ})) + \\ + W(\text{BE}) * M(\text{EZ}) =$$

$$= W(\text{BC}) * (\text{Sum}(\text{C}) + \text{Sum}(\text{E})) + \\ + W(\text{BD}) * (\text{Sum}(\text{D}) + \text{Sum}(\text{E})) + \\ + W(\text{BE}) * \text{Sum}(\text{E}) =$$

$$= W(\text{BC}) * \text{Sum}(\text{C}) + \\ + W(\text{BD}) * \text{Sum}(\text{D}) + \\ + W(\text{BE}) * \text{Sum}(\text{E})$$





# Это называется полукольцо

😊  
**Сочетательный закон (ассоциативность):**

**Сложение**

**Умножение**

$$(a+b)+c = a+(b+c)$$

$$(a*b)*c = a*(b*c)$$

**Переместительный закон (коммутативность):**

**Сложение**

$$a+b = b+a$$

**Нейтральный элемент:**

**Умножение**

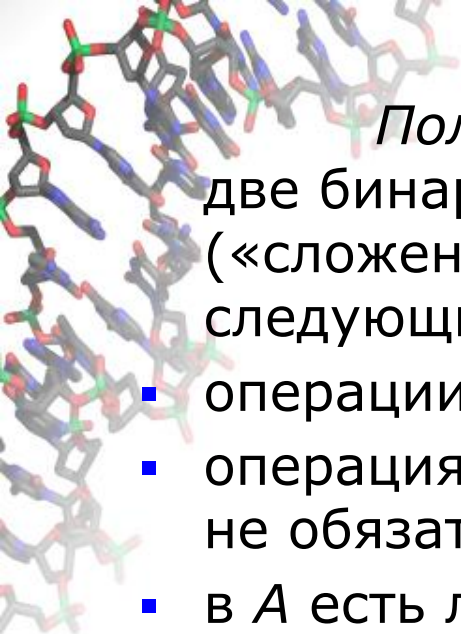
$$a*1 = 1*a = a$$

**РАСПРЕДЕЛИТЕЛЬНЫЙ ЗАКОН (ДИСТРИБУТИВНОСТЬ)**

**умножение относительно сложения**

$$(a+b)*c = a*c + b*c \quad a*(b+c) = a*b+a*c$$



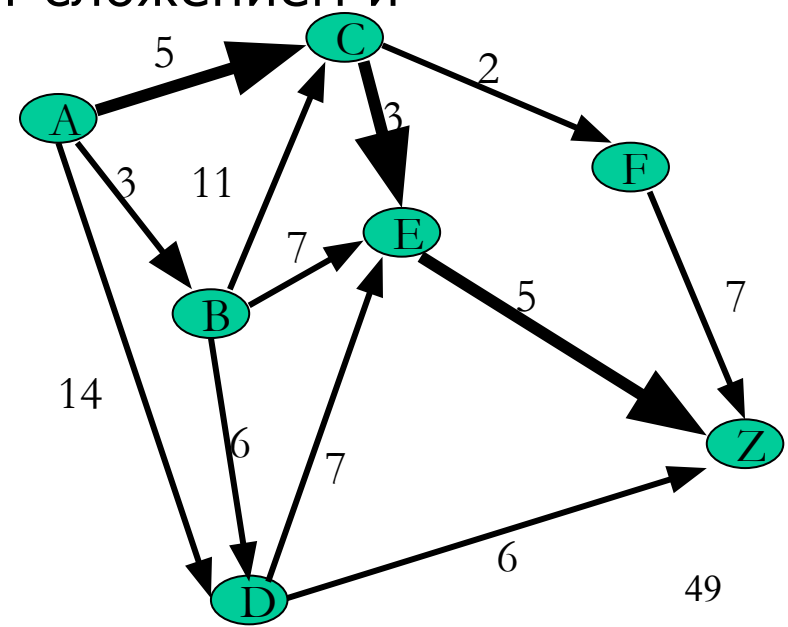


Полукольцо  $A$  – это множество, на котором заданы две бинарные всюду определенные операции  $+$  и  $*$  («сложение» и «умножение»), удовлетворяющие следующим свойствам:

- операции  $+$  и  $*$  ассоциативны;
- операция  $+$  коммутативна, коммутативность операции  $*$  не обязательна;
- в  $A$  есть левый нейтральный элемент относительно операции  $*$ ;
- Операции и обычно называют сложением и умножением.

$+$  - «целевая» операция

$*$  - «соединительная» операция



## Примеры полуколец.

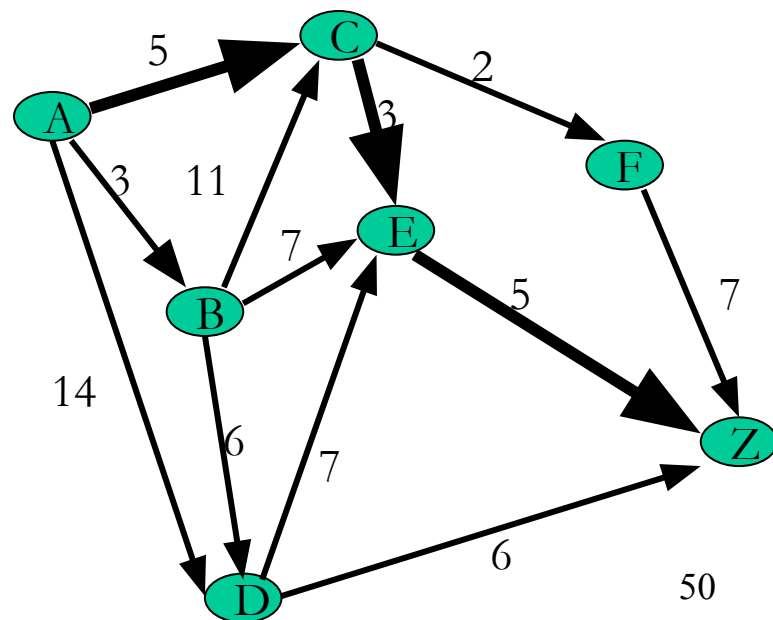
Первая операция – аналог сложения («целевая операция»), вторая – аналог умножения («соединяющая операция»):

- на числах:  $\{+, \times\}$ ,  $\{\min, +\}$ ;  $\{\max, \min\}$ ;
- на множествах:  $\{\cup, \cap\}$
- на множествах слов:  $\{\cup, \bullet\}$
- на матрицах:  $\{+, \times\}$ .

**+** - «целевая» операция

**\*** - «соединительная»

**операция**

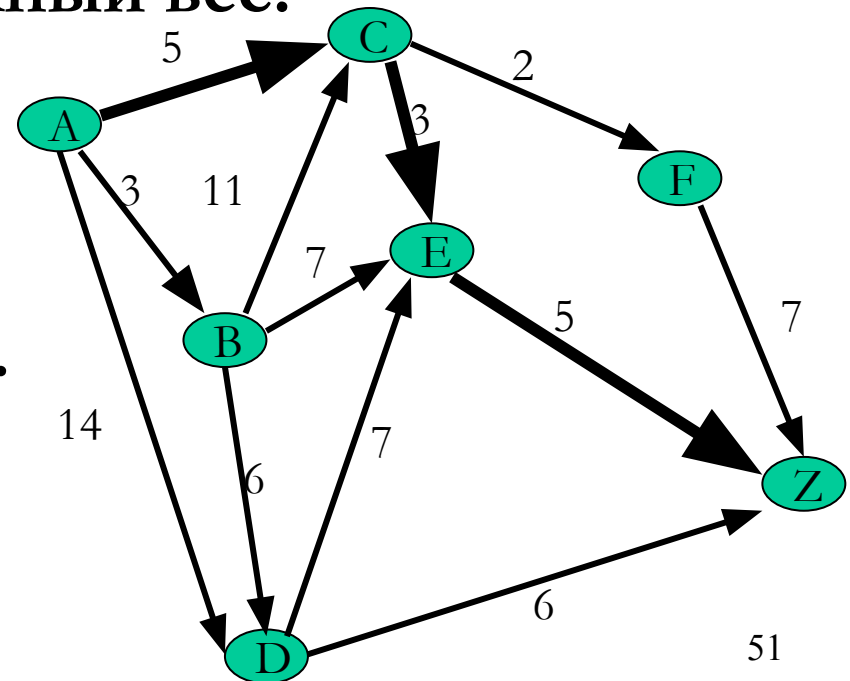


ДАНО: Ориентированный ациклический граф с весами на ребрах

$$G = \langle V, E, W; A, Z \rangle$$

ЗАДАЧА 1 Найти оптимальный полный путь, т.е. полный путь, имеющий минимальный (максимальный) возможный вес.

ЗАДАЧА 2 Найти сумму мультипликативных весов всех полных путей.





## Метод динамического программирования (Алгоритм Беллмана)

- Проход от стока к источнику:  
из  $W$  есть путь в  $V \Rightarrow$   
 $\Rightarrow W$  обрабатывается позже, чем  $V$ .
- Рекуррентное уравнение (минимальный путь)

$$\mathbf{BestW(A) = \min\{$$
$$\mathbf{W(AB) + BestW(B),}$$
$$\mathbf{W(AC) + BestW(C),}$$
$$\mathbf{W(AD) + BestW(D)}$$
$$\mathbf{\}}$$

- Рекуррентное уравнение (сумма  $m$ -весов):

$$\mathbf{Sum(A) =}$$
$$\mathbf{W(AB)*Sum(B) +}$$
$$\mathbf{+ W(AC)*Sum(C) +}$$
$$\mathbf{+ W(AD)*Sum(D)}$$
$$\mathbf{\}}$$

ДАННО: Ориентированный ациклический граф с весами на ребрах

$$G = \langle V, E, W; A, Z \rangle;$$

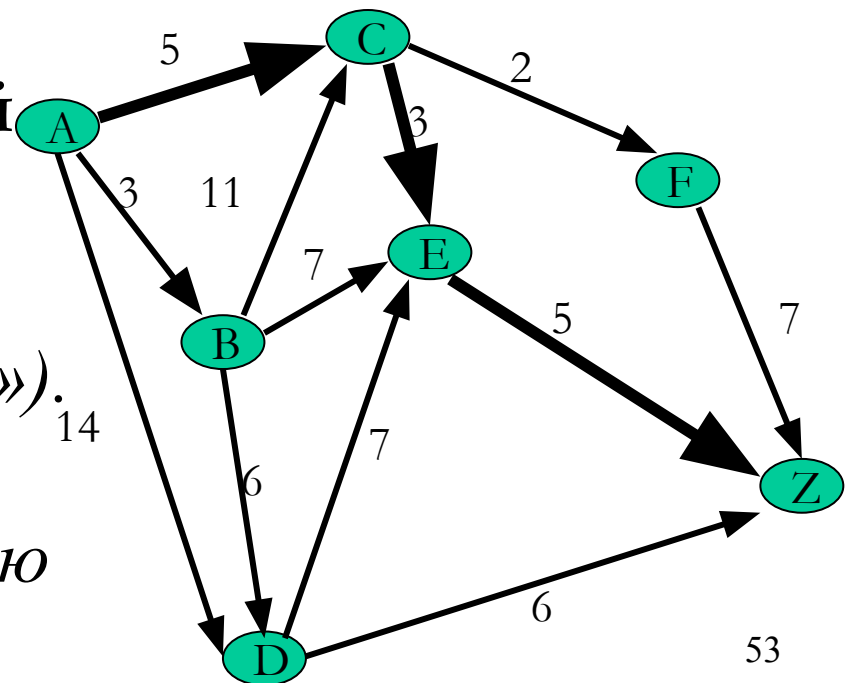
веса  $W(e)$  – элементы полукольца  $K$  с операциями  $+$  и  $*$ .

ЗАДАЧА 3 Найти сумму

мультипликативных

весов всех полных путей

Операция  $*$  («умножение») определяет веса путей («соединительная операция»).  
Операция  $+$  («сложение») определяет целевую функцию («целевая операция»).



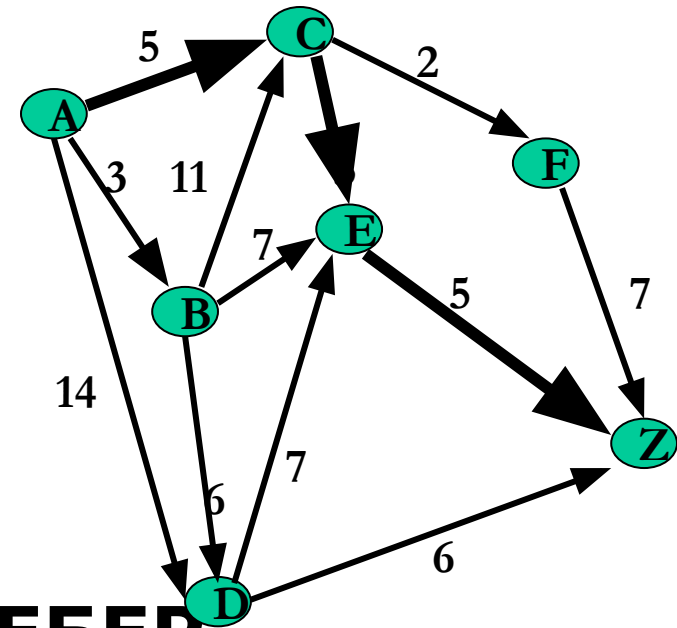


ДАНО: Ориентированный ациклический граф с весами на ребрах

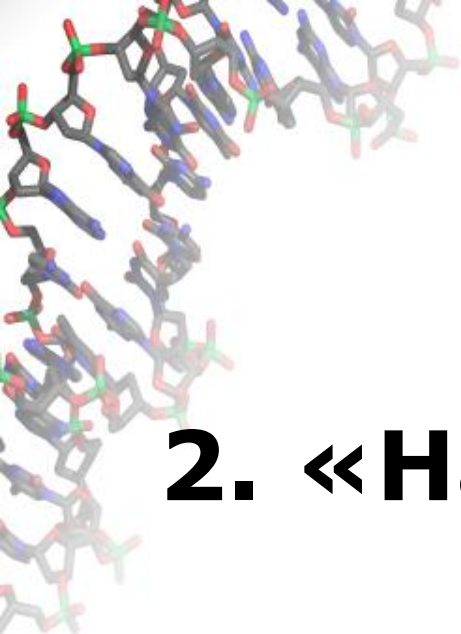
$$G = \langle V, E, W; A, Z \rangle;$$

веса  $W(e)$  – элементы полукольца  $K$  с операциями  $+$  и  $*$ .

ЗАДАЧА 3 Найти сумму мультипликативных весов всех полных путей.



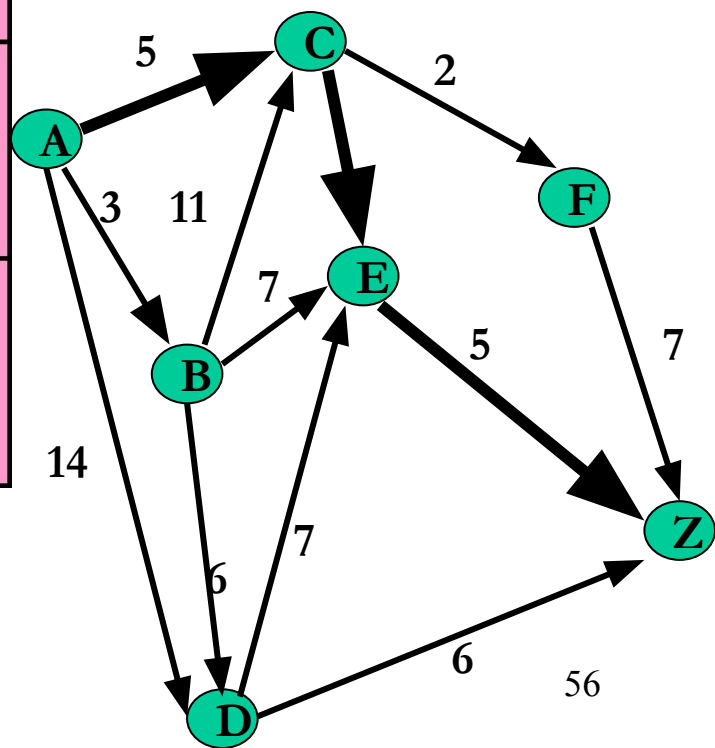
**ВРЕМЯ РАБОТЫ**  $\sim$  к-во РЕБЕР  
**ПАМЯТЬ**  $\sim$  к-во ВЕРШИН



## **2. «Накопление результата».**

Ранг	Вершина	Исх. ребра	Вес ребра	Вес след. верш.	Лучш вес для ребра	Лучш вес для верш	Куда идти
0	Z	xxx	0	0	0	0	xxx
1	F	Z	7	0	7	7	Z
1	E	Z	5	0	5	5	Z
2	D	E	7	5	12	6	Z
		Z	6	0	6		
2	C	E	3	5	8	8	E
		F	2	7	9		
3	B	C	11	8	19	12	D, E
		D	6	6	12		
		E	7	5	12		
4	A	B	3	12	15	13	C
		C	5	8	13		
		D	14	6	20		

$$\begin{aligned}
 \text{BestW}(B) &= \\
 &= \min\{ \\
 &\quad W(BC) + \text{BestW}(C), \\
 &\quad W(BD) + \text{BestW}(D), \\
 &\quad W(BE) + \text{BestW}(E), \\
 &\quad \}
 \end{aligned}$$



**Best Weight: 13**  
**Best Path: ACEZ**

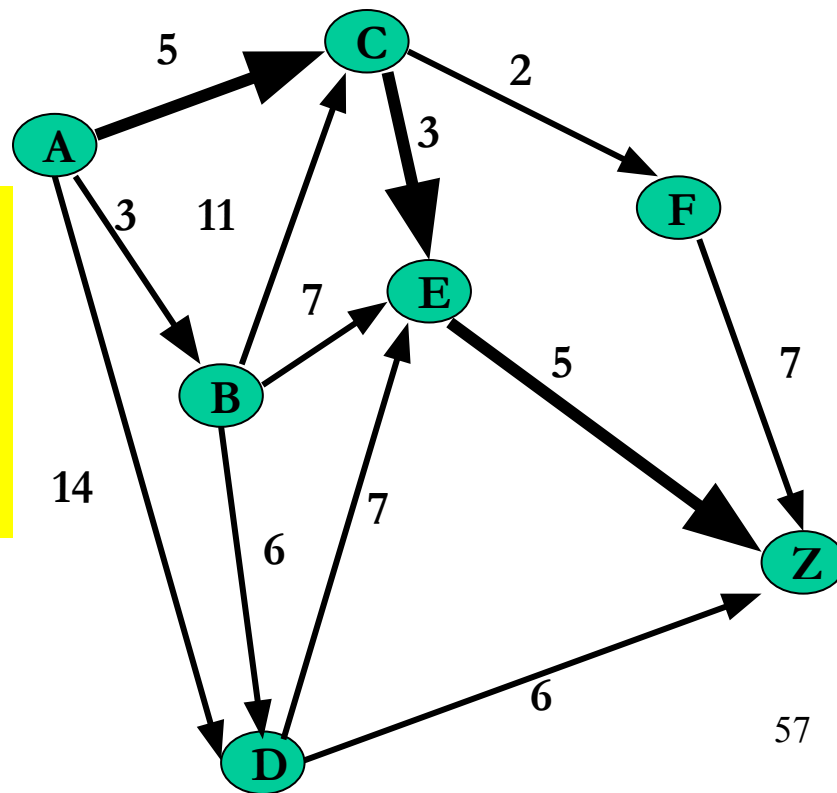
Верш.	Ранг	Куда
Z	0	xxx
D	?	Z, E
E	1	Z
F	1	Z

Ранг вершины T - это длина (количество ребер) в самом длинном пути из T в сток Z.

Обозначение: Rank(T)

Для любого T:  $Rank(T) = 1 + \text{Max} \{Rank(N_1), \dots, Rank(N_t)\}$   
 где  $N_1, \dots, N_t$  - наследники вершины T

$Rank(D) = 1 + \text{max} \{(Rank(E), Rank(Z))\}$



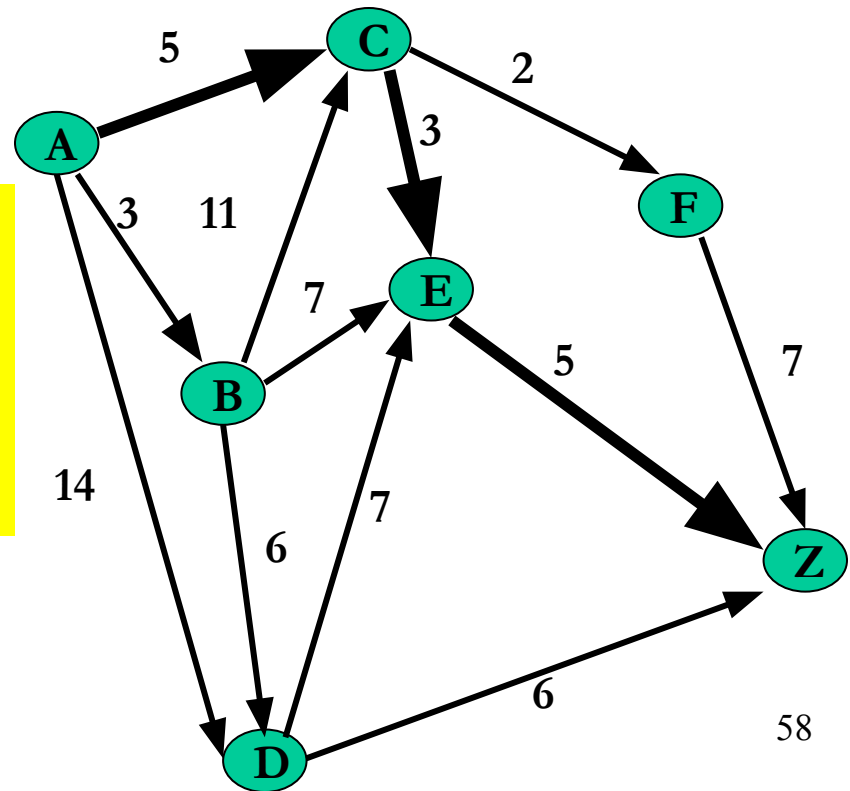
Ранг вершины  $T$  - это количество ребер в самом длинном пути из  $T$  в сток  $Z$ .

Обозначение:  $\text{Rank}(T)$

Упражнение 1.1.  
Найти ранги всех вершин данного графа  $G$ .

Для любого  $T$ :  $\text{Rank}(T) = 1 + \text{Max} \{ \text{Rank}(N_1), \dots, \text{Rank}(N_t) \}$   
где  $N_1, \dots, N_t$  - наследники вершины  $T$

$\text{Rank}(D) = 1 + \text{max} \{ (\text{Rank}(E), \text{Rank}(Z)) \}$

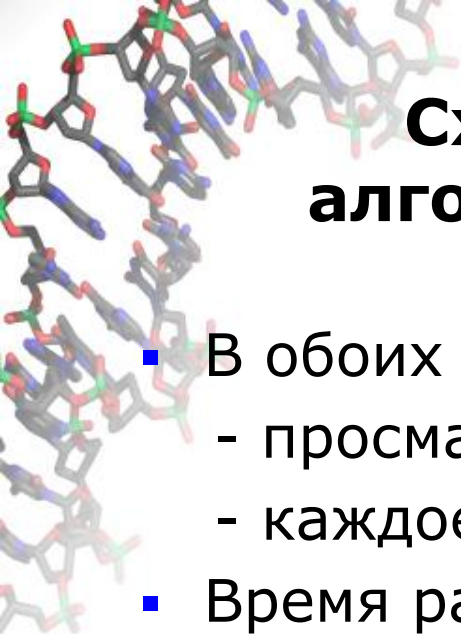




## Отличие алгоритма Беллмана от алгоритма с накоплением результата

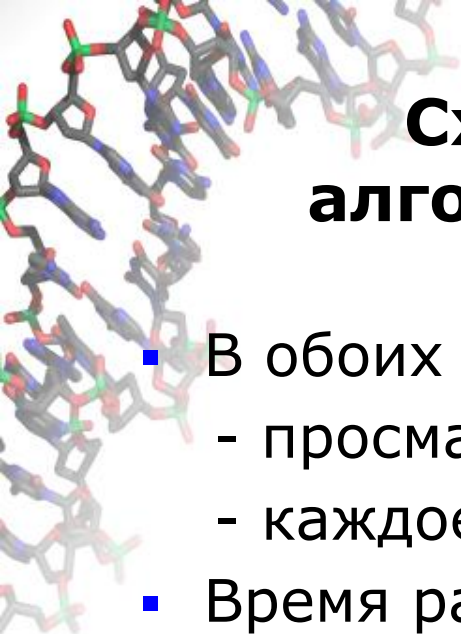
- АБ: Заранее известен топологический порядок на множестве вершин. Каждая вершина просматривается один раз, при этом результаты для наследников уже известны. Результат для данной вершины вычисляется при обработке вершины.
- НР: Порядок обработки вершин определяется динамически. Результат для вершины известен ДО начала ее обработки. При обработке вершины перевычисляются временные результаты для ее предшественников (идет накопление результата).





## Сходство алгоритма Беллмана и алгоритма с накоплением результата

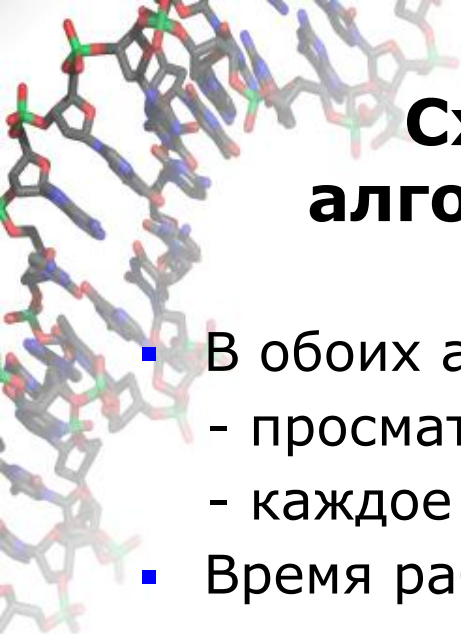
- В обоих алгоритмах:
  - просматриваются все ребра;
  - каждое ребро просматривается один раз.
- Время работы для обоих алгоритмов  
 **$T \sim \text{количество ребер}$**
- Расход памяти для обоих алгоритмов  
 **$M \sim \text{количество вершин}$**



## Сходство алгоритма Беллмана и алгоритма с накоплением результата

- В обоих алгоритмах:
  - просматриваются все ребра;
  - каждое ребро просматривается один раз.
- Время работы для обоих алгоритмов  
 **$T \sim \text{количество ребер}$**
- Расход памяти для обоих алгоритмов  
 **$M \sim \text{количество вершин}$**

ОБА АЛГОРИТМА ПРИМЕНИМЫ  
ДЛЯ ПРОИЗВОЛЬНЫХ ПОЛУКОЛЕЦ  
В отличие от алгоритма **Дейкстры [1959]** 😞



## Сходство алгоритма Беллмана и алгоритма с накоплением результата

- В обоих алгоритмах:
  - просматриваются все ребра;
  - каждое ребро просматривается один раз.
- Время работы для обоих алгоритмов

*$T \sim$  количество ребер*

- Расход памяти для обоих алгоритмов

*$M \sim$  количество вершин*

ОБА АЛГОРИТМА ПРИМЕНИМЫ  
ДЛЯ ПРОИЗВОЛЬНЫХ ПОЛУКОЛЕЦ

**В отличие от алгоритма Дейкстры** 😞

Для алг-ма Дейкстры требуется:

1) ( $<$ ,  $+$ )

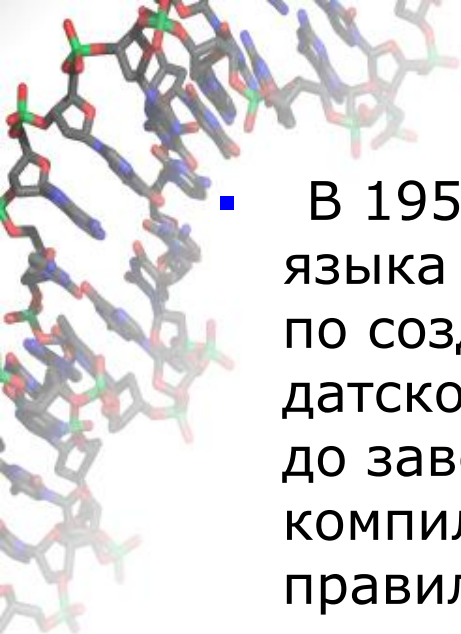
2)  $a < a+b$  - «дальше только хуже»



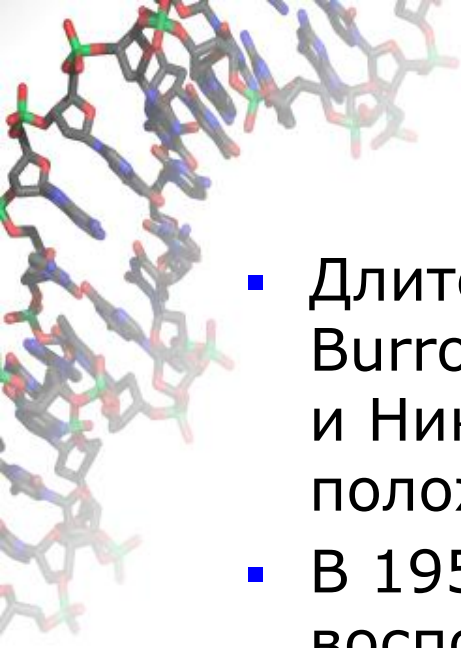
## **Эдсгер Вйбе Дэйкстра**

(нидерл. Edsger Wybe Dijkstra[3]  
(11 мая 1930, Роттердам,  
Нидерланды — 6 августа 2002,  
Нюэнен[nl], Нидерланды) —  
нидерландский учёный, труды  
которого оказали влияние на  
развитие информатики и  
информационных технологий;  
один из разработчиков  
концепции *структурного  
программирования*,  
*исследователь формальной  
верификации и распределённых  
вычислений*. Тьюринговский  
лауреат (1972).

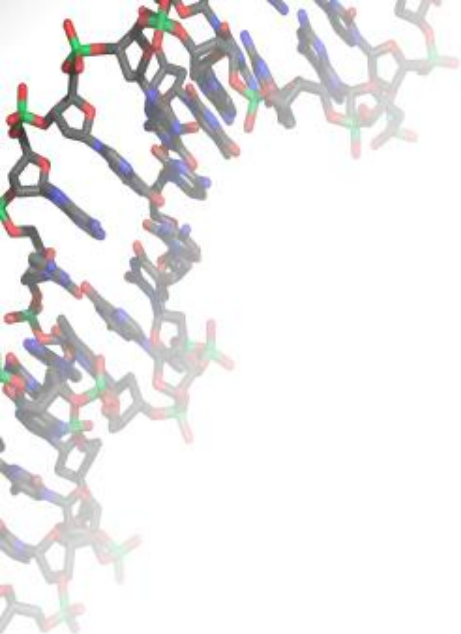




- В 1958—1960 годах принимал участие в разработке языка программирования Алгол, работал в команде по созданию компилятора языка; соревнуясь с датской командой Петера Наура, поклялся не бриться до завершения проекта и победил, написав компилятор за шесть недель, заодно изобретя новое правило компиляции — «вызов по имени».
- В 1960-е годы участвовал в создании операционной системы THE (англ.), построенной в виде множества параллельно исполняющихся взаимодействующих процессов. Именно в процессе этой работы появились понятия синхронизации процессов, идея семафора, а также была чётко осознана необходимость в структуризации процесса программирования и самих программ.

- 
- Длительное время работал в компании Burroughs. В 1970-е годы вместе с Тони Хоаром и Никлаусом Виртом разработал основные положения структурного программирования.
  - В 1957 году женился, по собственным воспоминаниям, в графе «профессия» анкеты, которую положено заполнять при бракосочетании, написал «программист» — и его заставили переписывать документы, заявив, что такой профессии не существует, в результате пришлось указать «физик-теоретик» [4].





**ЕСЛИ** графов **НЕ**  
хватает  
**ТО** что???  
**ИНАЧЕ** см. выше.

**3. ....**



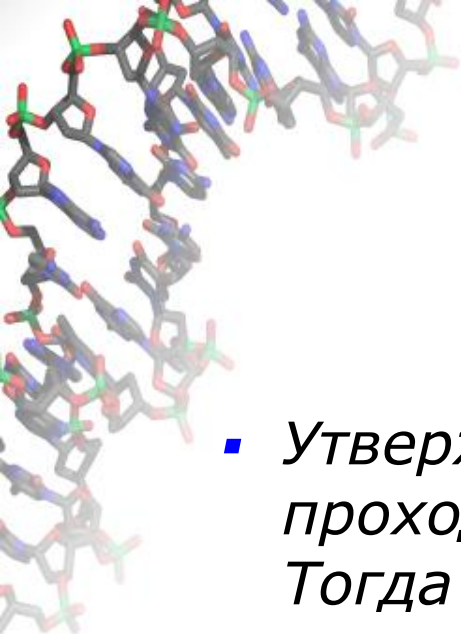
## Задача о дугах

**ДАНО:** набор из  $N$  дуг, каждая из которых соединяет две целочисленные точки числовой оси  $(x, y)$ , где  $1 \leq x < y \leq K$ . Все дуги расположены над числовой осью.

*Правильная система дуг* – это такая система дуг, в которой никакие дуги не пересекаются.

**НАДО:** найти в данном наборе правильную подсистему, содержащую максимально возможное количество дуг

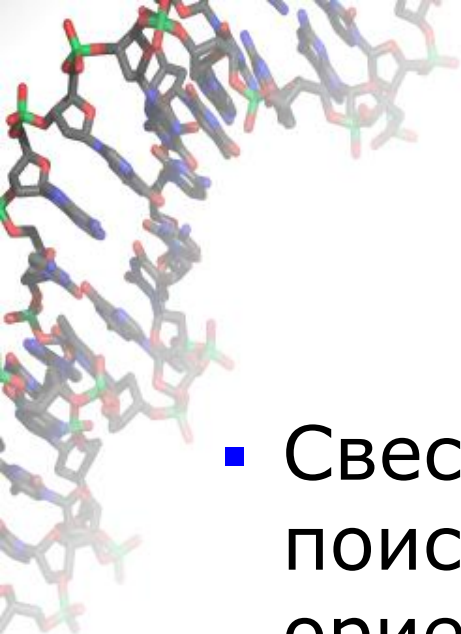
[Рисунок на доске]



- *Утверждение. Пусть в системе есть ровно  $k$  дуг, проходящих через точку  $1$ :  $(1, a_1), \dots, (1, a_k)$ . Тогда каждая правильная система принадлежит ровно к одному из следующих классов:*
- *0) не содержит дуги с концом в точке  $1$ ;*
- *1) содержит дугу  $(1, a_1)$ ;*
- *...*
- *$k$ ) содержит дугу  $(1, a_k)$ .*

## Решение

- Находим оптимальную систему для каждого отрезка  $[p, q]$ , где  $1 \leq p < q \leq K$ . Отрезки перебираем в порядке возрастания длин.
- Обозначение:  $M(p, q)$  – количество дуг в максимальной правильной подсистеме такой, что концы всех дуг в ней принадлежат отрезку  $[p, q]$ ,
- Рекуррентное уравнение:  $M(p, q) = \max$   
 $\{ M(p+1, q),$   
 $\max \{ 1 + M(p+1, t-1) + M(t+1, K) \mid$   
 $p < t \leq q \text{ И в наборе есть дуга } (p, t)$   
 $\}$



## *Задание*

- Свести задачу о дугах к задаче поиска максимального пути в ориентированном ациклическом графе



# Свести задачу о дугах к задаче поиска максимального пути в ориентированном ациклическом графе

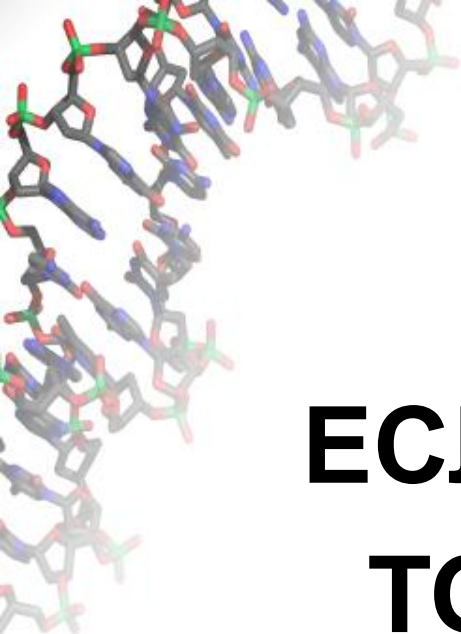
- Состояния – отрезки.
- Ребро соответствует сведению задачи для заданного отрезка к отрезку меньшей длины.
- Как определить ребра?





## Свести задачу о дугах к задаче поиска максимального пути в ориентированном ациклическом графе

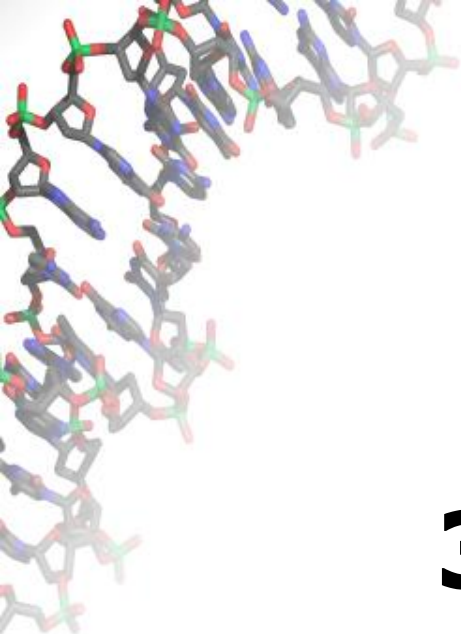
- Состояния – отрезки.
  - Ребро соответствует сведению задачи для заданного отрезка к отрезку меньшей длины.
  - Как определить ребра?
- НИКАК! Задача для отрезка  $[p, q]$  при включении в правильную систему дуги  $(p, t)$  сводится к решению **двух** задач – для отрезков  $[p+1, t-1]$  и  $[t+1, q]$ .



**ЕСЛИ графов НЕ хватает  
ТО *что???*  
ИНАЧЕ см. выше.**



**ЕСЛИ графов НЕ хватает  
ТО *ГИПЕРГРАФЫ*  
ИНАЧЕ см. выше.**



# **3. ГИПЕРГРАФЫ: ЗНАКОМСТВО И ПРИМЕРЫ**



# Графы и гиперграфы

Основные понятия-1. **Гиперребра и гиперпути.**

**Вершина**

**Гиперребро**

**Вершина-источник**

**Тупиковая вершина (сток)**

**Путь**

**Гиперпуть**

**Инициальный (гипер)путь**

**Терминальный (гипер)путь**

**Полный (гипер) путь:**

**Начальная вершина – источник;**

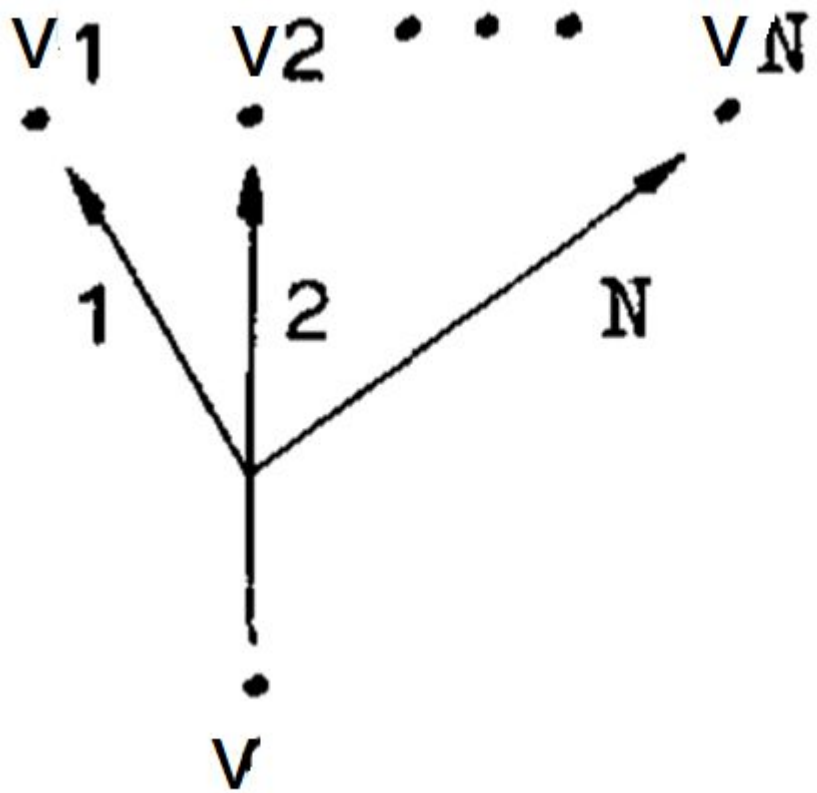
**Конечные вершины - тупиковые**

# (Ориентированный) гиперграф:

множество вершин и множество гиперребер

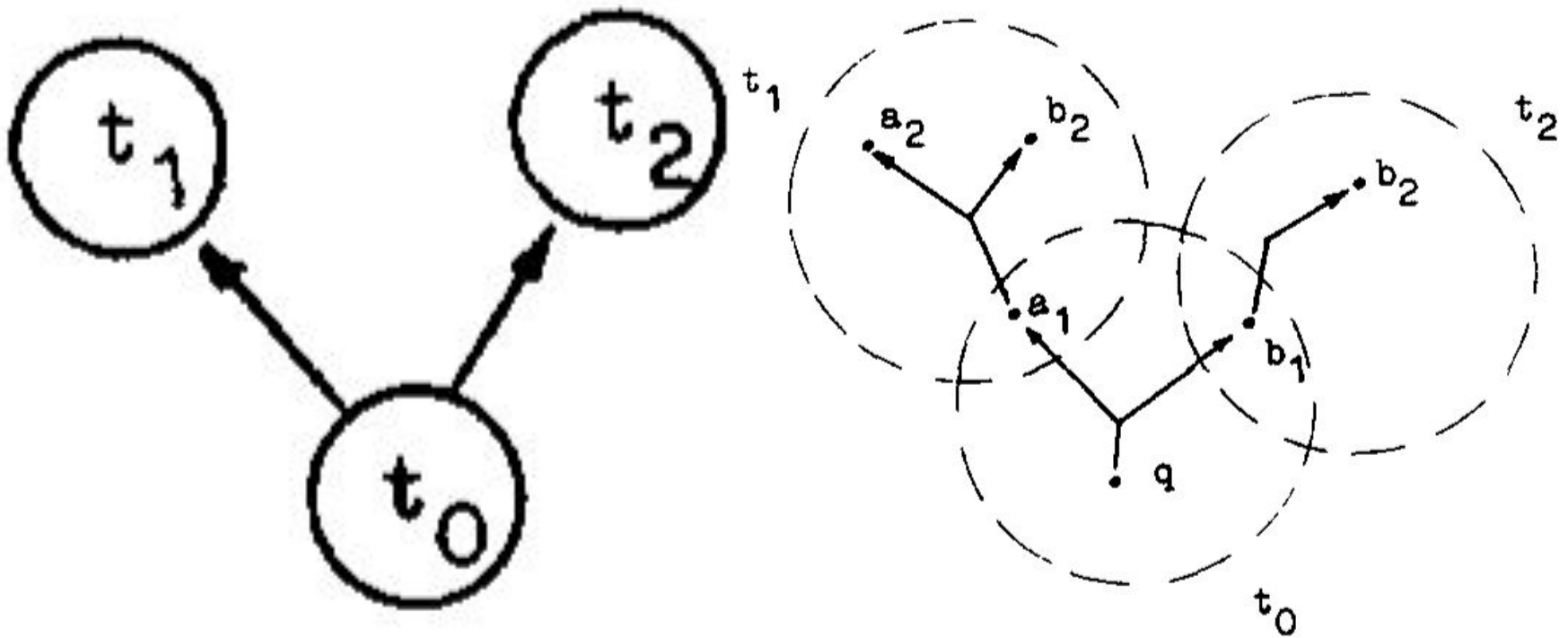
Гиперребро  $h = (v, (v_1, \dots, v_N))$

Ребро  $e = (v, v_1)$



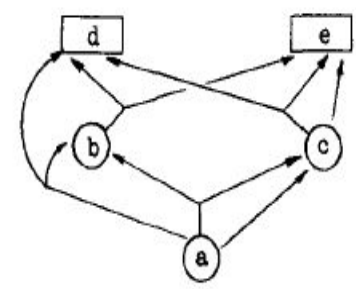


# Гиперпуть

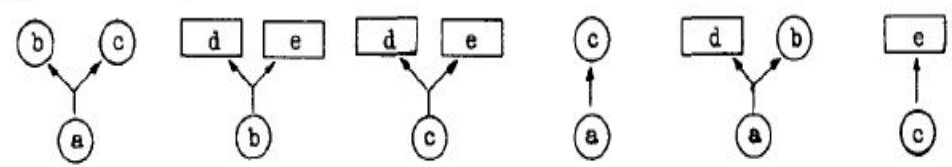


**Вес гиперпути – ПРОИЗВЕДЕНИЕ (\*)  
весов гиперребер**

(a)

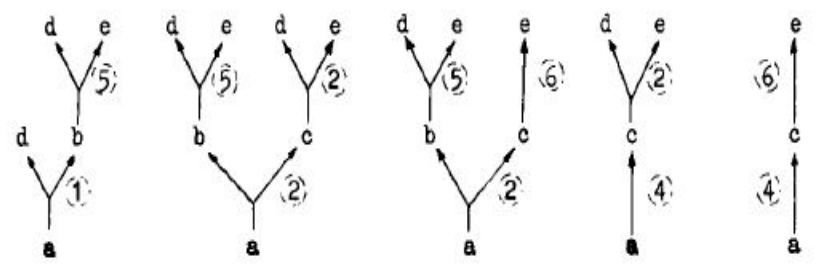


(b)



$U=(a,\{b,c\})$     $V=(b,\{d,e\})$     $W=(c,\{d,e\})$     $X=(a,\{c\})$     $Y=(a,\{d,b\})$     $Z=(c,\{e\})$   
 $r(U)=2$     $r(V)=5$     $r(W)=2$     $r(X)=4$     $r(Y)=1$     $r(Z)=6$

(c)



$(+,min):+:6$     $(+,min):+:9$     $(+,min):+:13$     $(+,min):+:6$     $(+,min):+:10$   
 $(x,+):x:5$     $(x,+):x:20$     $(x,+):x:60$     $(x,+):x:8$     $(x,+):x:24$



## Пример: задача о дугах

- Состояния – отрезки.
- **Гипер**-ребро соответствует сведению задачи для заданного отрезка к отрезку меньшей длины.
- Есть гиперребра двух типов:
  - 1) 0-гиперребра:  $\langle [p, q]; [p+1, q] \rangle$   
*В системе нет дуг с левым концом  $p$*
  - 2) дуговые гиперребра ( $(p, t)$  – дуга;  $t \leq q$ ):  
 $\langle [p, q]; [p+1, t-1], [t+1, q] \rangle$   
*Включаем в систему дугу  $(p, t)$*



# Веса. Повторение (полукольца).

Полукольцо с единицей  $A$  – это множество, на котором определены две бинарные всюду определенные операции  $+$  и  $*$ , удовлетворяющие следующим свойствам:

- операции  $+$  и  $*$  ассоциативны;
- операция  $+$  коммутативна,
- коммутативность операции  $*$  не обязательна;
- в  $A$  есть левый нейтральный элемент  $i$  относительно операции  $*$ ;
- операция дистрибутивна относительно операции  $*$  :  
$$a, b, c \in A ( (a + b) * c = (a * c) + (b * c) )$$
$$a, b, c \in A ( c * (a + b) = (c * a) + (c * b) )$$

Операции  $+$  и  $*$  обычно называют сложением и умножением.



# Графы и гиперграфы

## Основные понятия-2. **Веса.**

### *Вес гипер(ребра)*

*«Умножение»: как вычислять вес (гипер)пути*

*«Сложение»: целевая функция [коммутат.]*

*Дистрибутивность:*

$$a*(b+c) = a*b+a*c; (b+c)*a = b*a+c*a$$

*Вес пути*

*Вес гиперпути*

**ПРОБЛЕМА:**

**НАЙТИ «СУММУ» ВЕСОВ  
ВСЕХ ПОЛНЫХ (ГИПЕР)ПУТЕЙ**



## Пример: задача о дугах.

- Состояния – отрезки.
- **Гипер**-ребро соответствует сведению задачи для заданного отрезка к отрезку меньшей длины.
- Есть гиперребра двух типов:
  - 1) 0-гиперребра:  $\langle [p, q]; [p+s, q] \rangle$ ,  
где  $s$  – такое минимальное число, что в системе есть дуга  $(s, t); t \leq q$

*В системе нет дуг с левым концом  $p$*

**Вес гиперребра: 0**

- 2) дуговые гиперребра  $((p, t) - \text{дуга}; t \leq q)$ :  
 $\langle [p, q]; [p+1, t-1], [t+1, q] \rangle$

*Включаем в систему дугу  $(p, t)$*

**Вес гиперребра: 1**