21-MAVZU	**TYURING MASHINALARI. NOREKURSIV SANALUVCHAN TO‘PLAMLAR. TO‘XTASH MUAMMOSI. ALGORITMIK YECHILMAS MUAMMOLAR**

1. Tyuring mashinalari.

Agar qandaydir ommaviy muammoni yechish algoritmi ma’lum bo‘lsa, u holda uni realizatsiya etish uchun shu algoritmda aniq yoritilgan ko‘rsatmalarni ijro etish zarur. Algoritmni realizatsiya etish jarayonini avtomatlashtirish g‘oyasi, tabiiyki, inson bajaradigan ishni mashinaga uzatishni taqozo qiladi. Bunday mashinani XX asrning 30-yillarida amerika matematigi E.Post va angliya matematigi A.Tyuringlar tavsiya etdilar.

Tyuring mashinasining tushunchasi bizga intuitiv ma’lum bo‘lgan hisoblash protsedurasini elementar operatsiyalarga ajratish natijasida hosil bo‘ladi. Tyuring ta’kidlaydiki, istalgan mumkin bo‘lgan hisoblashni o‘tkazish uchun uning elementar operatsiyalarini qaytarish yetarli.

Tyuring ayrim turdagi nazariy hisoblash mashinasini izohlab berdi. Bu mashina muayyan mexanik qurilma emas, balki «xayoliy» matematik mashinadir. Berilgan ko‘rsatmani bajaruvchi hisoblovchi odamdan yoki mavjud raqamli hisoblash mashinasidan Tyuring mashinasi ikki jihati bilan farq qiladi.

Birinchidan, «Tyuring mashinasi» xato qila olmaydi, ya’ni u og‘ishmay (chetga chiqmasdan) ko‘rsatilgan qoidani bajaradi.

Ikkinchidan, «Tyuring mashinasi» potensial cheksiz xotira bilan ta’minlangan.

Endi Tyuring mashinasi tushunchasi bilan batafsil tanishamiz.

Tyuring mashinasini quyidagilar to‘liq aniqlaydi:

1.Tashqi alfavit, ya’ni $A=\{a_0, a_1, a_2, ..., a_n\}$ chekli simvollar to‘plami. A to‘plam elementlarining chekli ketma-ketligi A to‘plamdagi so‘z deyiladi. So‘zni tashkil etuvchi simvollar soni shu **so‘zning uzunligi** deyiladi.

Masalan, A alfavitning har bir elementi uzunligi 1 ga teng bo‘lgan so‘zdir. Bu alfavitda so‘z ko‘rinishida mashinaga beriladigan axborot (informatsiya) kodlashtiriladi. Mashina so‘z ko‘rinishida berilgan informatsiyani qayta ishlab, yangi so‘z hosil qiladi.

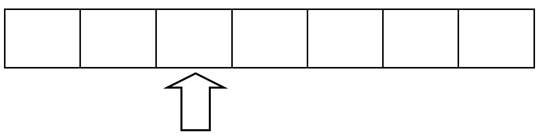
2.Ichki alfavit, ya’ni $q_0, q_1, q_2, ..., q_m, П, Л, Н$ simvollar. $q_0, q_1, q_2, ..., q_m$ - mashinaning chekli son holatlarini ifodalaydi. Istalgan mashinaning holatlari soni tayinlangan bo‘ladi. Ikki holatda maxsus vazifa bajariladi: q_1 - mashinaning boshlang‘ich (dastlabki) holati, q_0 - natijaviy (oxirgi) holati (to‘xtash holati). $П, Л, Н$ - surilish simvollaridir (o‘ngga, chapga va joyida).

3.Ikki tomonga cheksiz davom ettirish mumkin bo‘lgan lenta (mashinaning tashqi xotirasi). U katakchalarga (yacheykalarga) bo‘lingan bo‘ladi. Har bir katakchaga faqat bitta harf yozilishi mumkin. Bo‘sh katakchani a_0 simvoli bilan belgilaymiz (3.1-shaklga qarang).

a_0	a_2	a_3	a_3	a_7	a_9	a_{11}	a_{12}			

3.1-shakl.

4.Boshqaruvchi kallak (golovka). U lenta bo‘ylab harakat qiladi va qandaydir katakcha (yacheyka) qarshisida to‘xtashi mumkin (3.2-shakl).



3.2-shakl.

Bu holatda «kallak katakchani, ya’ni simvolni «ko‘rib turibdi»» deb aytamiz. Mashinaning bir takt davomidagi ishida kallak faqat bitta katakchaga surilishi (o‘ngga, chapga) yoki joyida turishi mumkin.

Lentada saqlanayotgan har bir informatsiya tashqi alfavitning a_0 dan farqli chekli simvollar majmuasi bilan tasvirlanadi.

Mashina ish boshlashidan oldin lentaga **boshlang‘ich axborot** (boshlang‘ich ma’lumot) beriladi. Bu holda boshqaruvchi kallak, qoidaga asosan, q_1 boshlang‘ich holatni ko‘rsatuvchi oxirgi chap belgi qarshisida turadi (3.3-shakl).

a_0	a_3	a_2	a_2	a_4	a_1	

q_1

3.3-shakl.

Mashinaning ishi taktlar yig‘indisidan iborat bo‘lib, ish davomida boshlang‘ich informatsiya oraliq informatsiyaga aylanadi.

Boshlang‘ich informatsiya sifatida lentaga tashqi alfavitning katakchalarga ixtiyoriy ravishda qo‘yilgan chekli simvollar sistemasini (alfavitdagi ixtiyoriy so‘zni) berish mumkin.

Berilgan boshlang‘ich informatsiyaga bog‘liq bo‘lgan ikki xil hol bo‘lishi mumkin:

1.Mashina chekli son taktdan keyin to‘xtaydi (q_0 to‘xtash holatiga o‘tadi). Bu vaqtda lentada B informatsiya tasvirlangan bo‘ladi. Bu holda mashina A boshlang‘ich informatsiyaga nisbatan tatbiq etiladigan (qo‘llanib bo‘ladigan) va uni qayta ishlab B natijaviy informatsiyaga keltirgan deb aytiladi.

2.Mashina hech vaqt to‘xtamaydi, ya’ni q_0 to‘xtash holatiga o‘tmaydi. Bu holda mashina A boshlang‘ich informatsiyaga nisbatan tatbiq etilmaydi deb aytiladi.

Mashina ishining har bir taktida quyidagi funksional sxema bo‘yicha harakat qiladi:

$$a_i q_j \to a_v \overset{П}{\underset{Н}{Л}} q_s\,.$$

Bu yerda a_i, a_v - tashqi alfavitning harflari; q_j, q_s - mashinaning holatlari; $П, Л, Н$ - surilish simvollari.

Boshqaruvchi kallak lentada qanday harfni ko‘rib turganligi (bizning yozuvda a_i) va mashina qaysi holatda (bizning yozuvda q_j) turganligiga qarab, bu taktda uch elementdan iborat komanda ishlab chiqiladi:

1) ko‘rib turilgan harf almashtirilgan **tashqi alfavit harfi** (a_v);

2) kelgusi takt uchun tashqi xotira adresi $\begin{pmatrix} П \\ Л \\ Н \end{pmatrix}$;

3) mashinaning kelgusi holati (q_s).

Hamma komandalar majmuasi **Tyuring mashinasining dasturini** tashkil qiladi. Dastur ikki o‘lchovli jadval shaklida bo‘lib, uni **Tyuring funksional sxemasi** deb ataydilar.

Bunday sxema 3.1-jadvalda misol sifatida berilgan.

3.1-jadval

	a_0	a_1	a_2
q_1	$a_2 л q_3$	$a_1 п q_2$	$a_2 л q_1$
q_2	$a_0 н q_2$	$a_2 н q_1$	$a_1 н q_2$
q_3	$a_0 п q_0$	$a_1 п q_4$	$a_2 н q_1$
q_4	$a_1 н q_3$	$a_0 п q_4$	$a_2 п q_4$

Aniqki, Tyuring mashinasining ishi butunlayiga uning dasturi bilan aniqlanadi. Agar ikkita Tyuring mashinalarining funksional sxemalari bir xil bo‘lsa, u holda ular bir-biridan farq qilmaydi. Har xil Tyuring mashinalari har xil dasturga ega bo‘ladi.

Bundan keyin Tyuring mashinasining har xil konfiguratsiyalarini (tarxiy ko‘rinishlarini) soddaroq ifodalash uchun lenta va uning katakchalarini ifodalamasdan axborotni faqat so‘z shaklida yozamiz. Boshqaruvchi kallak va mashina holatini ifodalash sifatida mashina holatini yozamiz.

3.1-jadvalda berilgan funksional sxemaga mos keluvchi Tyuring mashinasining ishini ko‘rib o‘taylik.

3.1-misol. Dastlabki konfiguratsiya quyidagicha berilgan bo‘lsin:

$$a_0 \quad a_2 \quad a_2 \quad a_0 .$$

$$q_1$$

Boshqaruvchi kallak a_2 harfini ko‘rib turganligi va mashina q_1 holatda bo‘lganligi uchun mashina $a_2 \, л \, a_2$ komandani ishlab chiqadi va natijada ikkinchi konfiguratsiyani hosil qilamiz

$$\begin{array}{cccc} a_0 & a_2 & a_2 & a_0 . \\ & q_1 & & \end{array}$$

Ravshanki, navbatdagi konfiguratsiyalar quyidagi ko‘rinishlarda bo‘ladi:

$\begin{array}{cccc} a_0 & a_2 & a_2 & a_0 \\ q_1 & & & \end{array}$ - uchinchi konfiguratsiya,

$\begin{array}{ccccc} a_0 & a_2 & a_2 & a_2 & a_0 \\ q_3 & & & & \end{array}$ - to‘rtinchi konfiguratsiya,

$\begin{array}{ccccc} a_0 & a_2 & a_2 & a_2 & a_0 \\ & q_0 & & & \end{array}$ - beshinchi konfiguratsiya.

Beshinchi konfiguratsiyada mashina q_0 holatda (to‘xtash holatida) turganligi uchun $a_2\, a_2\, a_2$ so‘z hisoblashning natijasi bo‘ladi. ■

4. Tyuring mashinasida algoritmni realizatsiya qilish

Bir qator misollarda ayrim oddiy arifmetik algoritmlarni realizatsiya etadigan Tyuring mashinasini qanday yasashni ko‘rsatamiz.

4.1-misol. Tyuring mashinasida o‘nlik sistemada n dan $n+1$ ga o‘tish algoritmini realizatsiya etish.

Yechim. O‘nlik sistemada n sonining yozuvi berilgan bo‘lsin va $n+1$ sonini o‘nlik sistemasidagi yozuvini ko‘rsatish talab etilsin, ya’ni $f(n) = n+1$ funksiyani hisoblash talab etilsin.

Ravshanki, mashinaning tashqi alfaviti 0,1,2,3,4,5,6,7,8,9 raqamlardan va bo‘sh katakcha a_0 dan iborat bo‘lishi kerak. Lentaga o‘nlik sistemada n sonini yozamiz. Bu yerda qatorasiga bo‘sh joysiz har bir katakchaga bitta raqam yoziladi.

Qo‘yilgan masalani yechish uchun ishning birinchi taktida mashina n sonining oxirgi raqamini o‘chirib, uni bir birlik katta songa almashtirib va agar oxirgi raqam 9 sonidan kichik bo‘lsa, u holda to‘xtash holatiga o‘tishi kerak.

Agar n sonining oxirgi raqami 9 bo‘lsa, u vaqtda mashina 9 raqamini o‘chirib, bo‘sh qolgan katakchaga 0 raqamini yozib, o‘sha holatda qolgan holda chapga yuqoriroq razryadli qo‘shnisiga surilishi kerak. Bu yerda ishning ikkinchi taktida mashina yuqoriroq razryadli raqamga 1 sonini qo‘shishi kerak.

Tabiiyki, chapga surilish paytida yuqoriroq razryadli raqam bo‘lmasa, u holda mashinaning boshqaruvchi kallagi bo‘sh katakchaga chiqishi mumkin. Bu holatda bo‘sh katakchaga mashina 1 raqamini yozadi.

Aytilganlardan shu narsa kelib chiqadiki, $f(n)=n+1$ funksiyani hisoblash algoritmini realizatsiya etish paytida mashina bor yo‘g‘i q_1 va q_0 holatlarda bo‘ladi.

Shunday qilib, o‘nlik sistemada n dan $n+1$ ga o‘tish algoritmini realizatsiya etadigan Tyuring mashinasi quyidagi ko‘rinishda bo‘ladi:

4.1-jadval

	a_0	0	1	2	3	4	5	6	7	8	9
q_1	1лq_0	1лq_0	2chq_0	3лq_0	4лq_0	5лq_0	6лq_0	7лq_0	8лq_0	9лq_0	0лq_1

1 va 2 – shakllarda $n=183$ va $n=399$ sonlar uchun mos ravishda konfiguratsiyalari keltirilgan.

$$a_0\underset{q_1}{18}3a_0 \qquad\qquad a_0\underset{q_1}{39}9a_0$$

$$a_0\underset{q_0}{18}4a_0 \qquad\qquad a_0\underset{q_1}{39}0a_0$$

$$a_0\underset{q_1}{30}0a_0$$

$$a_0\underset{q_1}{40}0a_0 \quad \blacksquare$$

4.1-shakl. 4.2-shakl.

5. Algoritmlar nazariyasining asosiy gipotezasi

Tyuring mashinasi algoritm tushunchasini aniqlashning bitta yo‘lini ko‘rsatadi. Shu tufayli bir nechta savollar paydo bo‘ladi: Tyuring mashinasi tushunchasi qanchalik umumiy bo‘ladi? Algoritmlarni Tyuring mashinasi vositasi bilan berish usulini universal usul deb bo‘ladimi? Hamma algoritmlarni shu usul bilan berish mumkinmi?

Ushbu savollarga hozirgi vaqtda mavjud bo‘lgan algoritmlar nazariyasi quyidagi gipoteza bilan javob beradi:

Har qanday algoritmni Tyuring funksional sxemasi orqali berish va mos Tyuring mashinasida realizatsiya etish (amalga oshirish) mumkin.

Bu gipoteza **Tyuring tezisi** deb aytiladi. Uni isbotlash mumkin emas, chunki bu tezis qat’iy ta’riflanmagan algoritm tushunchasini qat’iy aniqlangan Tyuring mashinasining tushunchasi bilan bog‘laydi.

Bu tezisni rad etish uchun Tyuring mashinasida realizatsiyalanmaydigan (amalga oshirilmaydigan) algoritm mavjudligini ko‘rsatish kerak.

Ammo hozirgacha aniqlangan hamma algoritmlarni Tyuring funksional sxemasi orqali realizatsiya etish mumkin.

Shuni ham ta’kidlab o‘tamizki, Markovning normal algoritm tushunchasi va Chyorch, Gyodel va Klinilar tomonidan kiritilgan rekursiv algoritm (rekursiv funksiyalar) tushunchalari Tyuring tomonidan kiritilgan algoritm tushunchasi (Tyuring funksional sxemasi) bilan ekvivalentligi isbotlangan.

Bu fakt o‘z navbatida Tyuring gipotezasining to‘g‘riligini yana bir marta ko‘rsatib o‘tadi.

1. Yechiluvchi va sanaluvchi to‘plamlar

Qandaydir alfavit berilgan bo‘lsin. Bu alfavitdagi hamma so‘zlar to‘plamini S bilan va S to‘plamning qism to‘plamini M bilan belgilaymiz.

1.1-ta'rif*. Agar x so'zning M to'plamga qarashlilik muammosini hal qila oladigan algoritm mavjud bo'lsa, u holda M ga yechiluvchi to'plam deb aytiladi.*

1.2-ta'rif. *Agar M to'plamning hamma elementlarini sanab chiqa oladigan algoritm mavjud bo'lsa, u holda M ga effektiv sanaluvchi to'plam deb aytiladi.*

1.1-teorema. *Agar M va L lar effektiv sanaluvchi to'plamlar bo'lsa, u holda $M \cup L$ va $M \cap L$ lar ham effektiv sanaluvchi to'plamlardir.*

Isbot. M va L lar effektiv sanaluvchi to'plamlar bo'lsin. U vaqtda, 2-ta'rifga asosan, ularning har biri uchun alohida algoritm mavjudki, ular orqali mos ravishda M va L dagi hamma elementlarni sanab chiqish mumkin. $M \cup L$ va $M \cap L$ to'plamlarning effektiv hisoblovchi algoritmi M va L to'plamlarning effektiv hisoblovchi algoritmlarini bir vaqtda qo'llash natijasida hosil qilinadi.

1.2-teorema **(Post teoremasi).** *M to'plamning o'zi va to'ldiruvchisi CM effektiv sanaluvchi bo'lsganda, shunda va faqat shundagina M to'plam yechiluvchidir.*

Isbot. **a)** M **to'plam va uning** CM **to'ldiruvchisi effektiv sanaluvchi bo'lsin.** U vaqtda, 2-ta'rifga asosan, bu to'plamlarning elementlarini sanab chiqa oladigan A va B algoritmlar mavjud bo'ladi. U holda M va CM to'plamlarning elementlarini sanab chiqish paytida ularning ro'yxatida x element uchraydi. Demak, shunday C algoritm yuzaga keladiki, u orqali x M to'plamga qarashlimi yoki qarashli emasmi degan muammoni hal qilish mumkin. Shunday qilib, M yechiluvchi to'plam bo'ladi.

b) M **yechiluvchi to'plam bo'lsin.** U holda, 1-ta'rifga asosan, x bu to'plamning elementimi yoki elementi emasmi degan muammoni hal qiluvchi algoritm mavjud bo'ladi. Bu algoritmdan foydalanib, M va CM to'plamlarga kiruvchi elementlarning ro'yxatini tuzamiz. Shunday qilib, M va CM to'plamlar elementlarini sanab chiquvchi ikkita A va B algoritmlarni hosil qilamiz.

Demak, M va CM to'plamlar effektiv sanaluvchi to'plamlar bo'ladi.

1.1-misol. $M = \{1, 4, 9, ..., n^2 ...\}$ natural sonlar kvadrat-lari to‘plami effektiv sanaluvchi to‘plam bo‘ladimi yoki yo‘qmi?

Yechim. $M = \{n^2\}$ to‘plam effektiv sanaluvchi to‘plam bo‘ladi, chunki uning elementlarini hosil etish uchun ketma-ket natural sonlarni olib, ularni kvadratga ko‘tarish kerak. Bu to‘plam ham yechiluvchi bo‘ladi. Haqiqatan ham, birorta x natural sonning M to‘plamga kirish yoki kirmasligini aniqlash uchun uni tub ko‘paytuvchilarga ajratish kerak. Bu usul uning natural sonning kvadratimi yoki yo‘qmi degan muammoni hal qilib beradi.

1.2-misol. Tartiblangan natural sonlar juftliklaridan iborat to‘plam effektiv sanaluvchi ekanligini isbotlang.

Yechim. Tartiblangan natural sonlar juftliklaridan iborat to‘plamning effektiv sanaluvchi ekanligini isbotlash uchun diagonal metodi deb aytiladigan metoddan foydalanamiz.

Buning uchun hamma tartiblangan natural sonlar juftliklarini quyidagi ko‘rinishda yozamiz:

$$\begin{array}{cccccc}
(0,0), & (0,1), & (0,2), & (0,3), & (0,4), & .... \\
(1,0), & (1,1), & (1,2), & (1,3), & (1,4), & .... \\
(2,0), & (2,1), & (2,2), & (2,3), & (2,4), & .... \\
(3,0), & (3,1), & (3,2), & (3,3), & (3,4), & .... \\
\multicolumn{6}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots}
\end{array}$$

Yuqori chap burchakdan boshlab ketma-ket diagonallar bo‘yicha o‘tib to‘plam elementlarini sanab chiqamiz. Bu juftliklarning ro‘yxati quyidagicha bo‘ladi:

(0,0), (1,0), (0,1), (2,0), (1,1), (0,2), (3,0), (2,1),
(1,2), (0,3), (4,0), (3,1), (2,2), (1,3), (0,4), ...

1.3-teorema. *Yechiluvchi bo‘lmagan effektiv sanaluvchi natural sonlar to‘plami mavjud.*

Isbot. Effektiv sanaluvchi ixtiyoriy U natural sonlar to‘plami berilgan bo‘lsin. U to‘plamning yechiluvchi emasligini isbotlash uchun, Post teoremasiga (1.2-teorema) ko‘ra, uning CU to‘ldiruvchisi effektiv sanaluvchi emasligini isbotlash yetarli.

$M_0, M_1, M_2, ...$ - hamma sanaluvchi natural sonlar to‘plamlaridagi effektiv sanab chiqilgan to‘plamlar bo‘lsin.

Demak, har qanday $n \in N$ uchun M_n to‘plamni tiklash mumkin.

Endi U to‘plamning hamma elementlarini sanab chiqadigan A algoritmni kiritaylik. Bu algoritm (m,n) raqamli qadamda $m \in M_n$ ni hisoblab chiqadi. Agar bu son n soni bilan ustma-ust tushsa, bu holda A algoritm uni U to‘plamiga kiritadi, ya’ni $n \in U \leftrightarrow n \in M_n$.

Bundan ko‘rinib turibdiki, har qanday sanaluvchi to‘plamdan CU to‘plam hech bo‘lmaganda bitta element bilan farq qiladi, chunki CU shunday n elementlardan iboratki, $n \bar{\in} M_n$. Shuning uchun ham CU sanaluvchi to‘plam emas. Demak, Post teoremasiga asosan U yechiluvchi to‘plam bo‘lmaydi.

Izoh. Isbot etilgan teorema aslida Gyodelning formal arifmetikaning to‘liqsizligi haqidagi teoremasini oshkormas (oshkora emas) ravishda qamrab olgan.

2. Markovning normal algoritmlari

2.1-ta’rif. *Bo‘sh bo‘lmagan chekli simvollar to‘plamiga alfavit va alfavitdagi simvollarga harflar deb aytiladi.*

2.2-ta’rif. A *alfavitdagi harflarning har qanday chekli ketma-ketligiga shu to‘plamdagi* ***so‘z*** *deb aytiladi. Harflarning bo‘sh ketma-ketligiga* ***bo‘sh so‘z*** *deb aytamiz va uni* $\wedge$ *simvol bilan belgilaymiz.*

Agar $S_{j_1} S_{j_2} ... S_{j_k}$ so‘zni P bilan va $S_{r_1} S_{r_2} ... S_{r_m}$ so‘zni Q bilan belgilasak, u holda $S_{j_1} S_{j_2} ... S_{j_k} S_{r_1} S_{r_2} ... S_{r_m}$ so‘z P va Q so‘zlarning birlashmasi $P\,Q$ ni bildiradi. Xususiy holda, $P\wedge = \wedge P = P$ va $(P_1 P_2) P_3 = P_1 (P_2 P_3)$.

Agar $B \subset A$ bo‘lsa, u vaqtda A ga B alfavitining kengayishi (kengaytirilgani) deb aytiladi. Ravshanki, bu holda B ning har bir so‘zi o‘z navbatida A alfavitining ham so‘zi bo‘ladi.

A alfavitidagi hamma so‘zlarning to‘plami D, C esa D to‘plamning biror qism to‘plami bo‘lsin, ya’ni $C \subset D$.

2.3-ta'rif. *Aniqlanish sohasi C va qiymatlar sohasi D bo'lgan effektiv hisoblanuvchi funksiyaga A alfavitdagi* ***algoritm (algorifm)*** *deb aytiladi.*

2.4-ta'rif. Agar *A alfavitdagi biror P so'z U algoritmning aniqlanish sohasiga tegishli bo'lsa, u holda U algoritm P so'zga* ***tatbiq etiladigan*** *deb aytiladi.*

2.5-ta'rif. *Agar $A \subset B$ bo'lsa, u holda B alfavitdagi har bir algoritm A* ***alfavit ustidagi algoritm*** *deb aytiladi.*

A alfavitdagi normal algoritm tushunchasi bilan A alfavit ustidagi normal algoritm tushunchasi o'rtasidagi farq juda ham muhimdir. A alfavitdagi har qanday normal algoritm faqat A ning harflaridan foydalanadi. A alfavit ustidagi normal algoritm bo'lsa A ga kirmagan boshqa qo'shimcha harflardan ham foydalanishi mumkin. Shunday qilib, A dagi har qanday normal algoritm A ustidagi normal algoritm ham bo'ladi. Ammo, A da shunday algoritmlar mavjudki, ular A ustida normal algoritm ekanligiga qaramasdan, A da normal algoritm bo'laolmay-dilar.

Ko'p aniqlangan algoritmlarni birmuncha oddiyroq qadamlarga bo'lish mumkin. Shu maqsadda rus matematigi A.A.Markov 1950 – yillarda algoritm tuzishning asosi (negizi) qilib, elementar operatsiya sifatida **bir so'zni ikkinchi so'z o'rniga qo'yishni** oladi.

Agar P va Q A alfavitdagi so'zlar bo'lsa, u holda $P \to Q$ va $P \to \cdot\, Q$ larni A alfavitdagi o'rniga qo'yish formulalari deb aytamiz. Bu yerda $\to$ va $\cdot$ simvollari A alfavitning harflari emas hamda P va Q larning har biri so'z bo'lishi mumkin. $P \to Q$ o'rniga qo'yish formulasi **oddiy** va $P \to \cdot\, Q$ o'rniga qo'yish formulasi **natijaviy (xulosaviy)** formula deb aytiladi.

Berilgan $P \to Q$ va $P \to \cdot\, Q$ o'rniga qo'yish formulalarining istalgan birini ifodalash uchun $P \to (\cdot)\, Q$ umumiy ko'rinishdagi yozuvni ishlatamiz.

Alfavitning quyidagi o'rniga qo'yish formulalarining chekli ro'yxati

$$P_1 \to (\cdot)\, Q_1$$

$$P_2 \to (\cdot)\, Q_2$$
....
$$P_r \to (\cdot)\, Q_r$$

algoritm sxemasi deb aytiladi va **u** A **alfavitda quyidagi algoritmni yuzaga keltiradi:**

Agar shunday W, V so‘zlar (bo‘sh so‘z bo‘lishlari mumkin) topilib, $Q = WTV$ bo‘lsa, u holda T so‘z Q so‘zning tarkibiga kiradi deb kelishib olamiz.

Endi A alfavitda P so‘z berilgan bo‘lsin. Bu yerda ikki hol bo‘lishi mumkin:

1. $P_1, P_2, ..., P_r$ so‘zlarning birortasi ham P so‘zning tarkibiga kirmaydi. Bu tasdiqni qisqa ravishda $U : P \supset$ shaklida yozamiz.

2. $P_1, P_2, ..., P_r$ so‘zlarning orasida P so‘zning tarkibiga kiruvchilari topiladi. Endi $1 \le m \le r$ munosabatni qanoatlantiruvchi eng kichik butun son m va P_m P ning tarkibiga kiruvchi so‘z bo‘lsin.

P so‘zning tarkibiga eng chapdan kirgan P_m so‘zni Q_m bilan almashtirishdan hosil bo‘ladiganni R deylik. P va R lar orasidagi aytilgan munosabatni:

a) agar $P \to (\cdot)\, Q_m$ o‘rniga qo‘yish formulasi oddiy formula bo‘lsa,

$$U : P| - R \qquad (2.1)$$

shaklida va

b) agar $P \to (\cdot)\, Q_m$ o‘rniga qo‘yish formulasi natijaviy formula bo‘lsa,

$$U : P| - \cdot R \qquad (2.2)$$

shaklida yozamiz.

(1) holda U algoritmi P so‘zini R so‘ziga oddiy o‘tkazadi deyiladi va (2.2) holda U algoritmi P so‘zini R so‘ziga natijaviy o‘tkazadi deb aytiladi.

$U : P| = R$ simvolik yozuv A alfavitda shunday $R_0, R_1, ..., R_k$ so‘zlar ketma-ketligi mavjudki, $P = R_0$, $R = R_k$, $J = 0,1,...,k-2$ lar uchun $U : R_j| - R_{j+1}$ va yoki $U : R_{k-1}| - R_k$, yoki $U : R_{k-1}| - \cdot R_k$ (oxirgi holda $U : P| = R$ o‘rniga $U : P| = \cdot R$ yoziladi) ekanligini bildiradi.

Yoki $U: P \models \cdot R$, yoki $U: P \models R$ va $U: R \supset$ bo‘lsa, shunda va faqat shundagina $U(P) = R$ deb qabul qilamiz.

Yuqoridagi kabi aniqlangan algoritmga normal algoritm yoki Markov algoritmi deb aytiladi.

U algoritmning amal qilishini quyidagicha ifodalash mumkin. A alfavitda P so‘z berilgan bo‘lsin. U algoritm sxemasida P_m so‘z P ning tarkibiga kiruvchi birinchi $P_m \to (\cdot)\, Q_m$ o‘rniga qo‘yish formulasini topamiz. P so‘zning tarkibiga eng chapdan kirgan P_m so‘z o‘rniga Q_m formulani qo‘yamiz. R_1 -shunday o‘rniga qo‘yishning natijasi bo‘lsin. Agar $P_m \to (\cdot)\, Q_m$ o‘rniga qo‘yish formulasi natijaviy bo‘lsa, u holda algoritmning ishi tugaydi va uning qiymati R_1 bo‘ladi. Agar $P_m \to (\cdot)\, Q_m$ o‘rniga qo‘yish formulasi oddiy bo‘lsa, u holda R_1 ga P ga nisbatan ishlatilgan protsedurani bajaramiz va hokazo. Agar oxirgi etapda $U: R_i \supset$ munosabatni qanoatlantiruvchi (ya’ni $P_1, P_2, ..., P_r$ so‘zlarning birortasi R_i tarkibiga kirmaydi) R_i so‘zi hosil bo‘lsa, u holda algoritmning ishi tugaydi va R_i uning qiymati bo‘ladi.

Agar ifodalangan jarayon oxirgi etapda tamom bo‘lmasa, u holda U algoritm P so‘zga tatbiq etilmaydi deb aytiladi.

2.1-misol. $\{b, c\}$ A alfaviti bo‘lsin. Quyidagi algoritm sxemasini ko‘ramiz

$$\left.\begin{array}{l} b \to \cdot \wedge \\ c \to c \end{array}\right\}.$$

Bu sxema bilan berilgan U normal algoritm A alfavitdagi tarkibiga kamida bitta b harfi kirgan har qanday P so‘zni shunday so‘zga o‘zgartiradiki, bu so‘z P so‘zdan uning tarkibiga eng chapdan kirgan b so‘zni o‘chirish natijasida hosil bo‘ladi.

Haqiqatan ham, P so‘zi tarkibiga eng chapdan kirgan b so‘zdan chaproqda turgan har qanday c harfni $c \to c$ oddiy o‘rniga qo‘yish formulasi yana c harfiga o‘tkazadi va eng chapdagi b harfini $b \to \cdot \wedge$ natijaviy o‘rniga qo‘yish formulasi $\wedge$ natijaviy bo‘sh so‘zga o‘zgartiradi.

Masalan, agar $P = ccbbc$ bo‘lsa, u holda $P \to \cdot Q$, bu yerda $Q = ccbc$. U algoritm bo‘sh so‘zni o‘z-o‘ziga o‘zgartiradi.

U algoritmi b harfi kirmagan bo‘sh bo‘lmagan so‘zlarga tatbiq etilmaydi. Haqiqatan ham, agar P so‘zi faqat c harflardan iborat bo‘lsa, u holda $c \to c$ oddiy o‘rniga qo‘yish formulasi uni yana o‘ziga aylantiradi. U vaqtda hamma vaqt $P \to P$ bo‘ladi va biz natijaviy o‘rniga qo‘yish formulasiga kelolmaymiz, ya’ni jarayon cheksiz davom etadi.

2.2-misol. $A-\{a_0, a_1, ..., a_n\}$ alfavit bo‘lsin. Quyidagi sxemani ko‘ramiz

$$\begin{cases} a_0 \to \wedge \\ a_1 \to \wedge \\ \vdots \\ a_n \to \wedge \end{cases}$$

Bu sxemani $\forall_i (a_i \to \wedge) \quad (a_i \in A)$ ko‘rinishida ham yozish mumkin. Bu sxema A alfavitdagi har qanday so‘zni bo‘sh so‘zga o‘zgartiradigan U normal algoritmdir.

Masalan, $U: a_1a_2a_1a_3a_0 |- a_1a_2a_1a_3 |- a_2a_1a_3 |- a_2a_3 |- \wedge$, va oxiri $U: \wedge \supset$. Demak, $U(a_1a_2a_1a_3a_0) = \wedge$.

2.3-misol. A alfavit S_1 harfdan iborat bo‘lsin. Bu harfni 1 bilan belgilaymiz. Har qanday n natural son uchun induksiya metodi bo‘yicha $\bar{0} = 1$ va $\overline{n+1} = \bar{n}1$ larni aniqlaymiz. Shunday qilib, $\bar{1} = 11$, $\bar{2} = 111$ va hokazo.

$\bar{n}$ so‘zlar raqamlar deb aytiladi. $\{\wedge \to \cdot 1$ sxema orqali berilgan U normal algoritmni aniqlaymiz.

A alfavitidagi har qanday P so‘z uchun $U(P) = 1P$ ga ega bo‘lamiz. Xususiy holda, har qanday n natural son uchun $U(\bar{n}) = \overline{n+1}$. Har qanday P so‘z bo‘sh so‘zi $\wedge$ ning kirishidan boshlanishini (chunki $P = \wedge P$) eslasak, keltirilgan algoritmning to‘g‘riligiga ishonamiz.

3. Markov bo‘yicha qismiy hisoblanuvchi va hisoblanuvchi funksiyalar.

U va K - algoritmlar va P - so‘z bo‘lsin. Agar U va K algoritmlarning ikkalasi ham P so‘zga tatbiq etilmaydigan yoki ikkalasi ham unga tatbiq etiladigan va keyingi holda $U(P) = K(P)$ bo‘lsa, bu holatni $U(P) \simeq K(P)$ ko‘rinishda ifodalaymiz.

Umuman, agar C va D - qandaydir ifodalar bo‘lsa, u holda $C \simeq D$ munosabat qo‘yidagini bildiradi: yoki ikkala ifoda ham aniqlanmagan, yoki ikkalasi ham aniqlangan va ular bir xil ob’ektni belgilaydi.

3.1-ta’rif. Agar A alfavitdagi istalgan P so‘z uchun $U(P) \simeq K(P)$ bo‘lsa, u holda U va K algoritmlar A ga nisbatan A alfavit ustida batamom (tamomila) ekvivalent deb aytiladi.

Agar P A alfavitdagi so‘z bo‘lganida har doim $U(P) \simeq K(P)$ hamda hech bo‘lmaganda $U(P)$ yoki $K(P)$ so‘zlarning birortasi aniqlangan va yana A ning so‘zi bo‘lsa, U va K algoritmlar A alfavitga nisbatan ekvivalent deb aytiladi.

$M - \{1, *\}$ alfavit, ω - hamma natural sonlar to‘plami, φ - n argumentli qismiy effektiv hisoblanuvchi arifmetik funksiya, ya’ni ω^n to‘plamning ayrim qism to‘plamini ω ga akslantiruvchi funksiya bo‘lsin.

B_φ orqali hech bo‘lmaganda bir tomoni aniqlangan holda har doim $B_\varphi(\overline{(k_1, k_2, ..., k_n)}) = \varphi\overline{(k_1, k_2, ..., k_n)}$ tenglikni o‘rinli qiladigan M dagi algoritmni belgilaymiz. Bu algoritm $\overline{(k_1, k_2, ..., k_n)}$ so‘zidan farq qiluvchi boshqa so‘zlarga tatbiq etilmaydi deb faraz qilamiz.

3.2-ta’rif. Agar M ustida M ga nisbatan B_φ ga batamom ekvivalent bo‘lgan normal algoritm mavjud bo‘lsa, u holda φ ga Markov bo‘yicha qismiy hisoblanuvchi funksiya deb aytamiz.

3.3-ta’rif. Agar φ funksiya har qanday n natural son uchun (hamma joyda) aniqlangan va Markov bo‘yicha qismiy hisoblanuvchi funksiya bo‘lsa, u holda unga Markov bo‘yicha hisoblanuvchi funksiya deb aytamiz.

Markov bo‘yicha qismiy hisoblanuvchi funksiya tushunchasi bilan qismiy rekursiv funksiya tushunchasi hamda Markov bo‘yicha hisoblanuvchi funksiya tushunchasi bilan umumrekursiv funksiya tushunchalari ekvivalentdir.

Keltirilgan tasdiqni isbotlovchi quyidagi teorema mavjud:

3.1-teorema. Har qanday qismiy rekursiv funksiya Markov bo'yicha qismiy hisoblanuvchi funksiya bo'ladi va har qanday umumrekursiv funksiya Markov bo'yicha hisoblanuvchi funksiyadir.

Quyidagi teoremalarni isbotsiz keltiramiz:

3.2-teorema. Agar A alfavit ustidagi U algoritm bo'yicha, ψ_U funksiya qismiy rekursiv (rekursiv) bo'lsa, u holda A alfavit ustida A ga nisbatan U algoritmga batamom ekvivalent bo'lgan normal algoritm mavjuddir.

3.3-teorema. Agar U A alfavit ustidagi normal algoritm bo'lsa, u holda ψ_U qismiy rekursiv funksiya bo'ladi; agar, bundan tashqari, U algoritm A alfavitdagi har qanday so'zga tatbiq etiladigan bo'lsa, u holda ψ_U umumrekursiv funksiya bo'ladi.

Natija. Agar berilgan φ qismiy funksiya Markov bo'yicha qismiy hisoblanuvchi funksiya bo'lsa, u qismiy rekursiv funksiya ham bo'ladi va agar φ Markov bo'yicha hisoblanuvchi funksiya bo'lsa, u holda φ umumrekursiv funksiya hamdir.

Natija va teoremalarning isboti E.Mendelson kitobining 242-244 va 246-249 betlarda keltirilgan.

Shunday qilib, keltirilgan natija va 1-teorema Markov bo'yicha qismiy hisoblanuvchi funksiya tushunchasi bilan qismiy rekursiv funksiya (xuddi shunday Markov bo'yicha hisoblash bilan rekursivlik) tushunchasining ekvivalentligini ko'rsatadi.

O'z navbatida Chyorch tezisi bo'yicha hisoblanuvchanlik tushunchasi bilan rekursivlik tushunchasi (qismiy effektiv hisoblanuvchanlik tushunchasi qismiy rekursivlik tushunchasiga) ekvivalentdir. A.A.Markov algoritmlar terminida normallashtirish (normalizatsiya) prinsipini yaratdi: A alfavitdagi har qanday algoritm A ga nisbatan A ustidagi biror normal algoritmga batamom ekvivalentdir.

Chyorch tezisi bilan normallashtirish prinsipining ekvivalentligi aniqlandi. Haqiqatan ham, Chyorch tezisi to'g'ri. A alfavitda U algoritm berilgan bo'lsin. Unga mos keladigan ψ_U funksiya qisman effektiv hisoblanuvchi bo‘ladi. U holda, Chyorch tezisiga asosan, ψ_U qismiy rekursiv funksiyadir. Demak, 2-teoremaga ko‘ra, U algoritm A ustidagi qandaydir normal algoritmga A ga nisbatan batamom ekvivalentdir. Shunday qilib, agar Chyorch tezisi to‘g‘ri bo‘lsa, u holda Markovning normallashtirish prinsipi ham to‘g‘ridir.

Endi normallashtirish prinsipi to‘g‘ri va φ ixtiyoriy qisman effektiv hisoblanuvchi funsiya, B_φ esa φ funksiyaga mos keluvchi M dagi algoritm bo‘lsin. Normallashtirish prinsipiga asosan B_φ algoritm M ustidagi biror normal algoritmga M ga nisbatan batamom ekvivalentdir. Demak, φ funksiya Markov bo‘yicha qisman hisoblanuvchi funksiyadir. U vaqtda olingan natijaga ko‘ra φ qisman rekursiv (rekursiv) funksiya bo‘ladi. Shunday qilib, Markovning normallashtirish prinsipidan Chyorch tezisini keltirib chiqardik.

Ma’lumki, algoritm va effektiv hisoblanuvchi funksiya tushunchalari intuitiv tushunchalar bo‘lganligi uchun biz Markovning normallashtirish prinsipi va Chyorch tezisining to‘g‘riligini isbot qilolmaymiz.

Shuni ham ta’kidlash kerakki, Chyorchning λ - hisoblanuvchanlik nazariyasi va Postning normal sistemalar nazariyasidan kelib chiqadigan tushunchalar ham qisman rekursiv funksiya yoki normal algoritm tushunchalariga ekvivalent bo‘ladi.

4. Algoritmik yechilmovchi muammolar

Matematika tarixida biror masalani yechish, odatda uning yechilish algoritmini topish deb hisoblanardi. Deyarli XX asr boshlarigacha hamma matematik masalalar algoritmik yechiluvchi masalalar deb qaralgan va ularni yechuvchi algoritmlar izlangan. Masalan, haqiqiy koeffitsientli n - darajali ko‘phadning ildizlarini uning koeffitsientlari yordamida radikallarda ifoda etish algoritmini izlash bir necha asrlar davom etdi. Masalan, uchinchi va to‘rtinchi darajali tenglamalar uchun bu algoritmni XVI asrda italyan matematiklari Kardano va Verarilar yaratdilar. Uzoq yillardan keyin norvegiyalik matematik Abel $n \geq 5$

bo‘lganda bunday algoritm mavjud emasligini ko‘rsatdi. Ikkinchi misol sifatida Gilbertning diofant tenglamalar haqidagi 10-muammosini ko‘rsatish mumkin. Bu muammoni Gilbert 1900 yilda Parijda e’lon qilgan edi. Deyarli 70 yildan keyin rus matematiklari Yu.Matiyasevich va G.Chudnovskiylar bu muammo algoritmik yechilmovchi muammo ekanligini isbotlab berdilar.

Faqat algoritmning intuitiv tushunchasidan Tyuring mashinasining aniq tushunchasiga o‘tish berilgan ommaviy muammoning algoritmik yechiluvchanlik masalasiga aniqlik kiritdi. Bu masalani quyidagicha ifodalash mumkin: berilgan ommaviy muammoni yechadigan Tyuring mashinasi mavjudmi yoki bunday mashina mavjud emasmi?

Bu savolga algoritmlar nazariyasi ayrim hollarda salbiy javob beradi. Shu turdagi natijalarni birinchilar qatorida 1936 yilda amerikalik matematik A.Chyorch oldi. U predikatlar mantiqidagi yechilish muammosi umumiy holda algoritmik yechimga ega emasligini ko‘rsatdi. O‘sha yilning o‘zida u matematik mantiqdagi keltirib chiqaruvchanlikni tanish muammosi ham algoritmik yechilmasligini isbot qildi. Keyingi masalani batafsilroq ko‘rib o‘taylik.

Matematikada aksiomatik metodning mazmuni quyidagidan iborat: berilgan nazariyaning hamma mulohazalari (teoremalari) shu nazariyada isbotsiz qabul qilingan mulohazalar (aksiomalar) dan formal mantiqiy keltirib chiqarish vositasi bilan olinadi.

Matematik mantiqda formulalarning maxsus tili ifodalanadi. U orqali matematik nazariyaning istalgan mulohazasi butunlay aniqlangan formula ko‘rinishida yoziladi. A asos (shart)dan B natijani mantiqiy keltirib chiqarish jarayonini dastlabki formulalarni formal almashtirishlar jarayoni sifatida ifodalash mumkin. Bunga mantiqiy hisobdan foydalanish yo‘li bilan erishi mumkin. Tanlangan mantiqiy hisobda B mulohazani A asosdan mantiqiy keltirib chiqarish masalasi A formuladan B formulaga olib keluvchi deduktiv zanjirning mavjudligi masalasidir. Shu tufayli keltirib chiqaruvchanlikni tanish muammosi paydo bo‘ladi: mantiqiy hisobdagi istalgan ikkita A va B formula uchun A dan Bga olib keluvchi deduktiv zanjir mavjudmi yoki yo‘qmi. Bu muammoning yechimi sifatida har qanday A va B lar uchun javob beradigan algoritm mavjudligi ma’nosida tushuniladi. Chyorch olgan natija quyidagicha izohlanadi:

4.1-Chyorch teoremasi. *Keltirib chiqaruvchanlikni tanish muammosi algoritmik yechilmovchidir.*

Tyuring mashinasining shifri tushunchasini kiritamiz. Hozirgacha Tyuring mashinasining dasturini ikki o‘lchovli $m \times n$ jadval ko‘rinishida yozardik. Ammo uni bir o‘lchovli shaklda ham tasvirlash mumkin. Buning uchun 5 ta simvollarni shunday ketma-ketlikda yozish kerakki, beshlikning birinchi simvoli jadvalning ustunini, ikkinchisi – satrini va keyingi uchta – jadvalning yuqorida ko‘rsatilgan ustun va satrlari kesishmasidagi uchta simvoldan (komandadan) iboratdir.

Masalan, - jadvalda ifodalangan funksional sxema o‘rniga quyidagi bir o‘lchovli satrni hosil qilamiz:

$$a_0 q_1 a_0 q_3 | q_1 \alpha \text{ н } q_2 \alpha \; q_1 \alpha \text{ л } q_1 \beta \; q_1 \beta \text{ л } q_1 a_0 q_2 a_0 \text{ л } q_4 \ldots \quad (4.1)$$

Mashina konfiguratsiyasini ifodalashda ham shu usuldan foydalanamiz, ya’ni holatni ifodalovchi harfni «ko‘rilayotgan» harfning tagidan emas, balki chap yonidan yozamiz. Masalan,

$$\underset{q_4}{|\,|\,|\,|\,|\,|}$$

konfiguratsiyani quyidagi shaklda yozamiz: $|\,|\,|\,|\, q_4 \,|\,|$.

(1) satrdagi har bir harfni quyidagi shartlarga rioya qilgan holda qayta nomlaymiz:

1) (1) satrni ayrim kodlashtirilgan guruhlarga bir qiymatli qilib bo‘lmoq kerak;

2) kod simvollari uch turda:

a) l, p, n harflari uchun;

b) tashqi alfavit harflari uchun;

v) mashina holatini ifodalovchi harflar uchun

bo‘lishlari kerak.

Shu munosabat bilan quyidagi kodlashtirish jadvalidan foydalanamiz:

Alfavit	Harf	Kodlashtirilgan guruh	Eslatmalar
Adreslar harfi	l	101	1lar orasida 1ta nol
	n	1001	1lar orasida 2ta nol
	p	10001	1lar orasida 3ta nol
Tashqi alfavit	a_0	100001 4ta nol	2dan katta juft sonli nollar
	a_1	10000001 6ta nol	
	....	....	
	a_n	10...01 2(n+2)ta nol	
Ichki alfavit	q_1	1000001 5ta nol	3dan katta toq sonli nollar
	q_2	100000001 7ta nol	
	....	....	
	q_m	10...01 2(n+1)+1 ta nol	

Agar (1) satrdagi $1,\alpha,\beta$ simvollarni mos ravishda a_1, a_2, a_3 harflar deb qarasak, u holda uni shu kodlashtirish sistemasi asosida quyidagicha yozish mumkin:

$$1000011000000110000110001100000000011000000011000001... \quad (4.2)$$

Funksional sxema yoki alohida olingan biror konfiguratsiya uchun tuzilgan 1 va 0 lardan iborat bo‘lgan bunday satrga **funksional sxemaning shifri** yoki **konfiguratsiyaning shifri** deb aytadilar.

Tyuring mashinasining lentasida mashina alfavitida yozilgan uning o‘z (xususiy) shifri tasvirlangan bo‘lsin.

Ikki hol bo‘lishi mumkin:

1.Mashina o‘zining shifriga tatbiq etiladi, ya’ni mashina bu shifrni qayta ishlaydi va chekli son qadamlardan so‘ng to‘xtaydi.

2.Mashina o‘zining shifriga tatbiq etilmaydi, ya’ni mashina hech qachon to‘xtash holatiga o‘tmaydi.

Shunday qilib, mashinalarning o‘zi (ularning shifri) ikki sinfga bo‘linadi: tatbiq etiladigan Tyuring mashinalari sinfiga va tatbiq etilmaydigan Tyuring mashinalari sinfiga. Shuning uchun quyidagi **ommaviy muammo:** o‘z-o‘ziga **tatbiq etiluvchanlik (samoprimenimost) ni tanish muammosi paydo bo‘ladi**.

Berilgan har qanday shifrga nisbatan shifrlangan Tyuring mashinasi qaysi sinfga kirishini aniqlash kerak: tatbiq etiladigan sinfgami yoki tatbiq etilmaydigan sinfgami?

4.2-teorema. *O‘z-o‘ziga tatbiq etiluvchanlikni tanish muammosi algoritmik yechimga ega emas (yechilmovchidir).*

Assotsiativlik hisobidagi so‘zlarning ekvivalentlik muammosi

Algoritmik hal etilmaslik haqidagi dastlabki natijalar matematik mantiq va algoritmlar nazariyalarida paydo bo‘lgan muammolar uchun olingan edi.

Ushbu muammolardan ayrimlarini 1-2 bandlarda keltirdik.

Keyinchalik shunga o‘xshash muammolar matematikaning turli xil qismlarida ham mavjud ekanligi aniqlandi. Shular qatoriga birinchi navbatda algebraik muammolar, shu jumladan, so‘zlar ekvivalentligi muammosidir.

$A=\{a,b,c,...\}$ alfavit va undagi so‘zlar to‘plamini ko‘rib o‘taylik. Agar L so‘zi M so‘zining bir qismi bo‘lsa, u holda L so‘zi M so‘zining tarkibiga kiradi deb aytamiz. Masalan, *bca* so‘zi *abcabac* so‘zining tarkibiga kiradi.

$$P-Q \text{ yoki } P\to Q$$

ko‘rinishidagi joiz o‘rniga qo‘yishlar orqali bir xil so‘zlarni ikkinchi xil so‘zlarga o‘zgartirishni ko‘rib o‘tamiz.

R so‘zining tarkibiga kamida bitta P so‘z kirgan bo‘lsagina, belgilangan yo‘nalishli (orientirlilangan) $P\to Q$ o‘rniga qo‘yishni R so‘ziga qo‘llash mumkin. Bu holda uning tarkibidagi istalgan bitta P so‘z Q so‘z bilan almashtiriladi.

Yo‘nalishsiz $P-Q$ o‘rniga qo‘yish R so‘ziga qo‘llash, uning tarkibidagi P so‘zni Q ga yoki Q so‘zini P ga almashtirishni bildiradi.

Asosan yo‘nalishsiz o‘rniga qo‘yishlarni ko‘rib o‘tamiz.

Misol. $ac-aca$ o‘rniga qo‘yishni $bcacab$ so‘ziga ikki xil usul bilan tatbiq etish mumkin: bu so‘z tarkibidagi aca so‘zini almashtirish $bcacab$ so‘zini va ac so‘zini almashtirish $bcacaab$ so‘zini beradi.

Bu o‘rniga qo‘yish formulasi $abcab$ so‘ziga tatbiq etilmaydi.

4.1-ta’rif. *Biror alfavitdagi hamma so‘zlar majmuasi bilan joiz o‘rniga qo‘yishlarning chekli sistemasiga (tizimiga) assotsiativ hisob deb aytiladi.*

Assotsiativ hisobni berish uchun unga mos bo‘lgan alfavit va o‘rniga qo‘yish sistemasini berish kifoya.

Agar R so‘zini joiz o‘rniga qo‘yishni bir marta tatbiq etish natijasida S so‘ziga almashtirish mumkin bo‘lsa, u vaqtda S ni ham shu yo‘sinda R ga almashtirish mumkin. Bu holatda R va S so‘zlariga **qo‘shni so‘zlar** deb aytiladi.

4.2-ta’rif. *Har bir* R_i *va* R_{i+1} $(i=1,2,...,n-1)$ *juft so‘zlar* $R_1, R_2, ..., R_{n-1}, R_n$ *so‘zlar ketma-ketligining qo‘shni so‘zlari bo‘lsa, u holda* $R_1, R_2, ..., R_{n-1}, R_n$ *so‘zlar ketma-ketligiga* R *so‘zidan* S *so‘ziga olib keladigan* ***deduktiv zanjir*** *deb aytamiz.*

Agar R so‘zidan S so‘ziga olib boradigan deduktiv zanjir mavjud bo‘lsa, S so‘zidan R so‘ziga olib boradigan deduktiv zanjir ham mavjud bo‘ladi. Bu holda R va S so‘zlariga **ekvivalent** so‘zlar deb aytamiz va $R \sim S$ ko‘rinishda belgilaymiz.

Har bir assotsiativ hisob uchun o‘zining maxsus **so‘zlar ekvivalentligi muammosi** mavjud:

berilgan assotsiativ hisobdagi har qanday ikkita so‘z uchun ular o‘zaro ekvivalentmi yoki yo‘qmi ekanligini bilish talab etiladi.

Assotsiativ hisob uchun so‘zlar ekvivalentligi muammosi 1911 yilda qo‘yilgan edi. O‘sha yilning o‘zida maxsus ko‘rinishdagi ba’zi assotsiativ hisoblar uchun so‘zlar ekvivalentligini tanish (raspoznaniya) algoritmi tavsiya etilgandi.

Tabiiyki, istalgan assotsiativ hisobga tatbiq etilishi mumkin bo‘lgan umumiy algoritmni topish masalasi vujudga keldi.

1946 va 1947 yillarda bir-biridan bexabar holda rus matematigi A.Markov va amerika matematigi E.Post **istalgan assotsiativ hisobi uchun so‘zlar ekvivalentligini tanish algoritmi mavjud emasligini ko‘satdilar**. Ular shunday muayyan assotsiativ hisoblar tuzdilarki, ularning har biri uchun so‘zlar ekvivalentligi muammosi algoritmik yechilmovchi edi.

1955 yilda rus matematigi P.S.Novikov guruhlarning aynan tengligi muammosi algoritmik yechimga ega emasligini isbotladi. Bu muammo formal ravishda assotsiativ hisobidagi so‘zlar ekvivalentligi muammosining xususiy holidir.

A.Markov va E.Post tomonidan tadqiq etilayotgan muammoning algorimik yechimga ega emasligini ko‘rsatish uchun tuzilgan misollar ancha murakkab va yuzdan ortiq joiz o‘rniga qo‘yishlar qo‘llanilgandi.

Sankt-Peterburglik matematik G.S.Seytlin shu muammoning algoritmik yechilmovchiligini isbotlash uchun tuzgan misolida faqatgina yettita joiz o‘rniga qo‘yishdan foydalanadi.

Navbatdagi misol sifatida predikatlar mantiqidagi yechilish muammosi va diofant tenglamalar to‘g‘risidagi Gilbertning 10-muammosini ko‘rsatish mumkin.

1936 yilda amerikalik matematik A.Chyorch predikatlar mantiqidagi yechilish muammosi umumiy holda algoritmik yechilmovchiligini isbotladi.

1970 yilda rus matematiklari Yu.V.Matiyasevich va G.V.Chudnovskiylar diofant tenglamalar haqidagi Gilbertning 10-muammosi algoritmik yechimga ega emasligini ko‘rsatganliklarini yana bir eslatamiz.

Shunday qilib, matematikada ko‘plab ommaviy muammolar algoritmik yechimga ega emas.