Latin négyzetek alkalmazásai a tervezésben és kódolásban

DÉNES TAMÁS

tdenest@freemail.hu

Kulcsszavak: vizsgálat-optimalizálás, titkosítás, adatbiztonság, képkódolás

A latin négyzeteket a gyakorlatban főleg három területen alkalmazzák. Ezek a következők: a statisztikus kísérlet-tervezés, a kódolás (hírközlési alkalmazásként) és a titkosítás.

Míg a statisztikai alkalmazásokban R. A. Fisher tekinthető úttörőnek, az 1920-as években megkezdett tevékenységével és híres könyvével (The design of experiments), addig a latin négyzetek alkalmazása a hírközlésben a II. világháborút megelőzően kezdődött mind amerikai, mind német részről. Az első publikált eredmények azonban, érthető okokból, csak a háború befejezése után jelentek meg. Amerikai részről C. E. Shannon, német részről pedig Rudolf Schauffler nevét kell megemlíteni úttörőként, aki a német rejtjelfejtés kimagasló alakja volt a II. világháború alatt.

1. Alkalmazások a statisztikus kísérlet tervezésben

R. A. Fisher szerint – szemben az addigi gyakorlattal, amikor is a kísérletek során csak egyetlen tényezőt változtattak – célszerű az összes tényező egyidejű variálása. Így jött létre a statisztika egy új ága, a faktor-analízis, valamint a latin négyzetek kísérletek tervezésében való felhasználása.

Az ortogonális latin négyzetek statisztikai alkalmazását egy példán keresztül szemléltetem. Öt különféle kikészítésű szálból szőtt késztermék mintázatát kell minőségileg összehasonlítani. A késztermék előállításán öt szövőgépen öt gépkezelő dolgozik. Az a feltételezés (amit igazolni kell), hogy a szálak kikészítésén kívül a szövéshez felhasznált gép és kezelője is minőséget befolyásoló tényezők. Ha a kísérletekre szánt idő nem lenne korlátozó tényező, akkor minden egyes fonalfajtát mind az öt szövőgépen az öt gépkezelő mindegyikével ki kellene próbálni. Ez összesen 125 kísérletet jelent. A latin négyzetek segítségével azonban kielégítő eredményt lehet elérni egy 25 kísérletből álló kísérletsorozattal.

Tegyük fel, hogy

K_1, $K_{2,...,}$ K_5 jelöli az öt gépkezelőt,
S_1, $S_{2,...,}$ S_5 jelöli az öt szövőgépet, valamint
Y_1, $Y_{2,...,}$ Y_5 az öt különféle szál jelölésére szolgál.

A minőség összevetésére szolgáló 25 kísérletet az *1. ábrán* bemutatott latin négyzet szemlélteti.

Az *1/a. ábrán* látható latin négyzet úgy alkalmazható a kísérletek tervezésénél, hogy az oszlop kiválasztással a kezelőt, a sor kiválasztással a szövőgépet és a kiválasztott sor és oszlop metszetében álló elemmel az adott kísérletben felhasznált fonalat határozzuk meg. Így például az első kísérlet: a K_1 gépkezelő az S_1 szövőgépen Y_1 fonallal dolgozik.

	K_1	K_2	K_3	K_4	K_5
S_1	Y_1	Y_4	Y_5	Y_2	Y_3
S_2	Y_3	Y_1	Y_2	Y_4	Y_5
S_3	Y_2	Y_5	Y_1	Y_3	Y_4
S_4	Y_5	Y_3	Y_4	Y_1	Y_2
S_5	Y_4	Y_2	Y_3	Y_5	Y_1

1/a. ábra

1	4	5	2	3
3	1	2	4	5
2	5	1	3	4
5	3	4	1	2
4	2	3	5	1

1/b. ábra

Tegyük fel továbbá, hogy a gépkezelők hatékonyságát befolyásolja, hogy a hét mely munkanapján dolgoznak. Akkor az 1/a. ábrán megadott latin négyzetben szereplő indexekhez (lásd *1/b. ábra*) tartozó *ortogonális párt* kell szerkeszteni (lásd *2/a. ábra*), ahol a munkanapokat számok jelzik. (1=hétfő, 2=kedd, 3= szerda, 4=csütörtök, 5=péntek.)

	K_1	K_2	K_3	K_4	K_5
S_1	1,1	4,2	5,3	2,4	3,5
S_2	3,3	1,4	2,1	4,5	5,2
S_3	2,2	5,3	1,5	3,4	4,1
S_4	5,5	3,1	4,3	1,2	2,4
S_5	4,4	2,5	3,2	5,1	1,3

2/a. ábra

1	2	3	4	5
3	4	1	5	2
2	3	5	4	1
5	1	3	2	4
4	5	2	1	3

2/b. ábra

A két ortogonális latin négyzet (lásd az 1/b. és 2/a. ábrán) egymásra helyezésével megszerkesztett 25 kísérletből álló kísérletsorozatot ábrázolja a *2/b. ábra,* amely lehetővé teszi, hogy minden egyes gépkezelő minden egyes szövőgépen dolgozzon, a munkájában az öt különböző kikészítésű fonal mindegyikét pontosan egyszer használja és a vele kapcsolatos kísérleteknél egy hét 5 munkanapja közül minden napra egy kísérlet jusson.

Hasonló típusú kísérletek megtervezése merül fel például a növénytermesztés, vagy a gyógyszerkutatás területén is.

Nyilvánvaló, hogy a kísérlet tervezésénél a latin négyzetek alkalmazhatósága bizonyos szempontból korlátozott, hiszen ha az előbbi példánkban például a gépkezelők száma nem öt, hanem négy, akkor már másfajta elrendezésre van szükség. Az ilyen, a latin négyzeteknél általánosabb elrendezéseket *block designoknak* nevezzük. Az érdeklődő olvasó jó betekintést kaphat a block designokról [6]-ből.

Egy másik példa világítja meg a *teljes latin négyzetek alkalmazását* a kísérletek tervezésében.

Egy állatkísérletben a kísérleti állatokat különböző étrend szerint táplálják, feltevés szerint (amit a kísérletek során ellenőrizni kívánnak) egy adott állat etetése előtt, a kísérlet során kapott összes étkezések száma, valamint a közvetlenül megelőző etetés során kapott takarmány fajtája befolyásolja az eredményt.

Tegyük fel, hogy *n* darab állat és *n* féle takarmány kerül a kísérletben felhasználásra. n=4 esetén az A_1, A_2, A_3, A_4 kísérleti állatot a T_1, T_2, T_3, T_4 takarmányokkal táplálják a *3. ábrán* látható teljes latin négyzet szerinti kísérleti elrendezésben.

	1	2	3	4
A_1	T_1	T_2	T_3	T_4
A_2	T_2	T_4	T_1	T_3
A_3	T_3	T1	T_4	T_2
A_4	T_4	T_3	T_2	T_1

3. ábra

Az ábrán látható kísérleti elrendezés azt jelenti, hogy például az A_1 állatnak első étkezésre T_1 takarmányt, másodikra T_2 takarmányt stb. kell adni. A kísérletsorozatban valóban fontos a teljes latin négyzet tulajdonság (ennek definíciója e cikk első részében található), hiszen ez biztosítja, hogy az összes lehetséges takarmány-sorrendet kipróbáljuk, ami éppen a kísérlet egyik lényeges célját képezi.

2. Kódolás-elméleti alkalmazások

Érdekes megemlíteni, hogy az általánosan elterjedt nézetekkel szemben, nem Richard Wesley Hamming volt az első, aki 1950-ben a hibajelző és javító kódokat bevezette (lásd a jobboldali *képet*), hanem Rudolf Schauffler akinek latin négyzetek alapján szerkesztett nem bináris hibajelző és javító kódjai, valamint a latin négyzetek egyéb alkalmazásai már az 1946-ban benyújtott doktori diszszertációjában szerepeltek.

Mivel Schauffler eredményei a titkosszolgálatnál töltött évei és a szigorú titoktartás következtében, szélesebb körben hozzáférhető módon csak 1956-ban jelentek meg, így Hamming bináris hibajelző és javító kódjait független eredménynek kell tekinteni. Schauffler gondolatait jóval később viszontláthatjuk [1,5].

A latin négyzeteken alapuló nem bináris hibajelző és javító kódok elterjedéséhez a feltételt azonban a szélessávú űrtávközlési csatornák megjelenése teremtette meg. Ezért S. W. Golomb és E. C. Posner a JPL (Jet Propulation Laboratory) pasadénai kutatólaboratórium vezető munkatársaiként foglalkoztak a hibajelző és javító kódok latin négyzetek alapján való szerkesztésével. Eredményüket a következőkben lehet megfogalmazni:

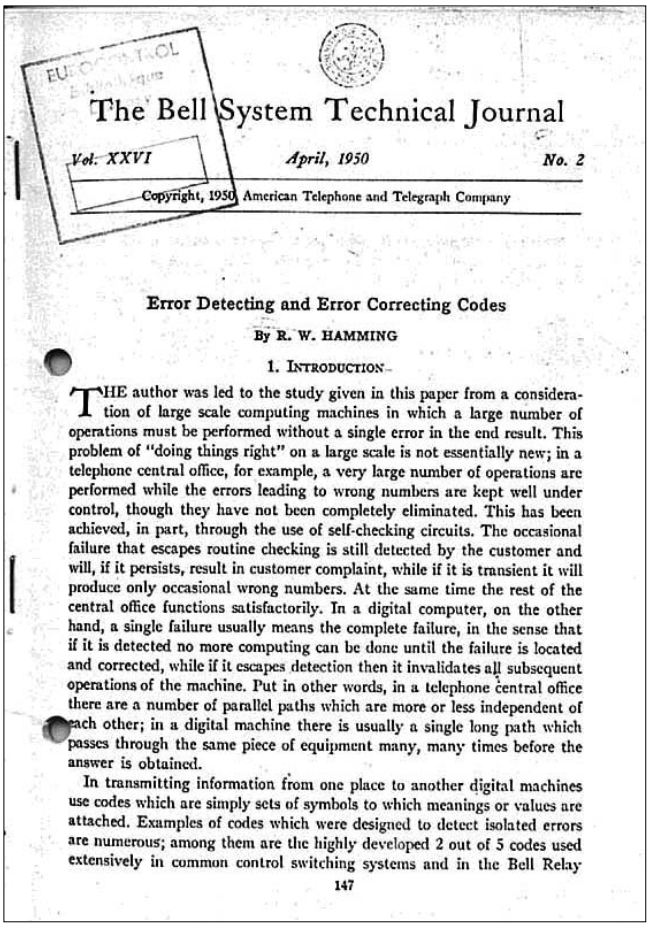
Ha létezik *t* darab n-ed rendű latin négyzetből álló ortogonális rendszer, akkor létezik olyan *t+2* hosszúságú kódszavakból álló kód, amelynek minimális Hamming-távolsága *t+1* és amelyben n^2 kódszó van.

Két *k* hosszúságú kódszó
$a=(a_1, a_2 \ldots\ldots a_k)$ és $b=(b_1, b_2 \ldots\ldots b_k)$ közötti
Hamming távolság (jele = *d(a,b)*),
azon *i* (i=1,2,……k) indexeknek a száma,
amelyekre $a_i \neq b_i$ teljesül.

Egy kódnak a *minimális Hamming-távolsága* a kódban szereplő összes kódszó párok közötti Hamming - távolságok minimuma.

Alapvető tételként kell e helyen megemlíteni, hogy egy *q* betűből álló ábécé feletti *k* hosszúságú *d* minimális Hamming-távolságú kódban maximum q^{k-d+1} kódszó lehet.

Így a fent leírt Golomb-Posner kódban a kódszavak száma $n^{t+2-(t+1)+1}=n^2$, ami n-ed rendű latin négyzetek esetén maximális.



The Bell System Technical Journal

Vol. XXVI April, 1950 No. 2

Error Detecting and Error Correcting Codes

By R. W. HAMMING

1. INTRODUCTION

THE author was led to the study given in this paper from a consideration of large scale computing machines in which a large number of operations must be performed without a single error in the end result. This problem of "doing things right" on a large scale is not essentially new; in a telephone central office, for example, a very large number of operations are performed while the errors leading to wrong numbers are kept well under control, though they have not been completely eliminated. This has been achieved, in part, through the use of self-checking circuits. The occasional failure that escapes routine checking is still detected by the customer and will, if it persists, result in customer complaint, while if it is transient it will produce only occasional wrong numbers. At the same time the rest of the central office functions satisfactorily. In a digital computer, on the other hand, a single failure usually means the complete failure, in the sense that if it is detected no more computing can be done until the failure is located and corrected, while if it escapes detection then it invalidates all subsequent operations of the machine. Put in other words, in a telephone central office there are a number of parallel paths which are more or less independent of each other; in a digital machine there is usually a single long path which passes through the same piece of equipment many, many times before the answer is obtained.

In transmitting information from one place to another digital machines use codes which are simply sets of symbols to which meanings or values are attached. Examples of codes which were designed to detect isolated errors are numerous; among them are the highly developed 2 out of 5 codes used extensively in common control switching systems and in the Bell Relay

147

A Golomb-Posner kód konstrukciója a következő példán jól követhető:

Legyen n=4, a konstrukcióhoz az L_1, L_2, L_3 páronként ortogonális negyedrendű latin négyzeteket használjuk fel *(4. ábra)*.

$L_1 =$

	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

$L_2 =$

0	1	2	3
2	3	0	1
3	2	1	0
1	0	3	2

$L_3 =$

0	1	2	3
3	2	1	0
1	0	3	2
2	3	0	1

4. ábra

Látható módon L_1-hez a keret elemeket is feltüntettük, mivel ennek a kódszavak képzése során jelentősége lesz. A kódszavakat a következő módon képezzük:

Az első komponens a sorkeret elem a_i (i=1,2,3,4), a második komponens az oszlopkeret elem b_j (j=1,2,3,4), majd sorrendben ezen keret elemek után következnek az L_1, L_2, L_3 latin négyzetek belsejében a_i sor b_j oszlop metszésénél lévő elemek. Így a 4. ábra L_1, L_2, L_3 latin négyzeteiből az *5. ábrán* lévő kódszó készlethez jutunk.

Az olvasó ellenőrizheti, hogy a kódszavak száma $4^2=16$, a szóhossz *5*, a minimális Hamming-távolság *4*.

(0 0 0 0 0)	(1 0 1 2 3)	(2 0 2 3 1)	(3 0 3 1 2)
(0 1 1 1 1)	(1 1 0 3 2)	(2 1 3 2 0)	(3 1 2 0 3)
(0 2 2 2 2)	(1 2 3 0 1)	(2 2 0 1 3)	(3 2 1 3 0)
(0 3 3 3 3)	(1 3 2 1 0)	(2 3 1 0 2)	(3 3 0 2 1)

5. ábra

Az első részben említett tized rendű latin négyzetekből álló ortogonális rendszer létrehozásának problémája (létezik-e három 10-ed rendű latin négyzetből álló ortogonális rendszer?) most a kódok nyelvére lefordítva így hangzik: van-e olyan Golomb-Posner kód, melyben a 10 elemű ábécé feletti kódszavak száma 100, hosszuk 5, és a kód minimális Hamming-távolsága 4?

A Golomb-Posner kódok előnye, hogy $n \neq 2$, illetve $n \neq 6$ esetén tetszőleges n elemű ábécé felett léteznek. [2]-ben sikerült a Golomb-Posner féle konstrukciót ortogonális latin téglalapokra általánosítani

A *latin téglalap* olyan téglalap mátrix, amely kiegészíthető latin négyzetté. Két azonos méretű *latin téglalapot akkor nevezünk ortogonálisnak*, ha egymásra helyezve a megfelelő rendezett párok mind különbözőek.

Példát mutatunk be a párok [2]-ban közölt konstrukciója alapján arra, hogy az R_1, R_2, R_3, R_4, R_5 *2x6* méretű latin téglalapokból álló ortogonális rendszerből *(6. ábra)* milyen kód nyerhető.

6. ábra

$R_1 =$

	1	2
1	6	1
2	1	2
3	2	3
4	3	4
5	4	5
6	5	6

$R_2 =$

6	2
1	3
2	4
3	5
4	6
5	1

$R_3 =$

6	3
1	4
2	5
3	6
4	1
5	2

$R_4 =$

6	4
1	5
2	6
3	1
4	2
5	3

$R_5 =$

6	5
1	6
2	1
3	2
4	3
5	4

A konstrukció követhetősége érdekében R_1-nél a perem elemeket is feltüntettük. A konstrukció a latin négyzetekre alkalmazott Golomb-Posner eljárásnak értelemszerű analogonja. Az így kapott kód elemeit (kódszavait) a *7. ábrán* láthatjuk.

(1166666)	(4133333)
(1212345)	(4245612)
(2111111)	(5144444)
(2223456)	(5256123)
(3122222)	(6155555)
(3234561)	(6261234)

7. ábra

A fenti konstrukció felhasználható személyi számok, jogosítvány, vagy ISBN számok, valamint más hasonló kódok előállítására. Az ortogonális rendszer miatt a keletkező kódszavak garantáltan különbözőek, az eljárás könnyen programozható, gyors előállítást kínál.

3. Alkalmazások a távközlésben

Egy mobiltelefon hálózati rendszerében egy nagyobb területet felosztanak kisebb területi egységekre és minden egységnek van egy telepített adó-vevő központja. Ez a központ a környezetében dolgozó adók üzenetét veszi és továbbítja egy másik központ felé. Természetesen a területi központ a más körzetből érkező üzeneteket veszi és a saját területi egységén belül osztja szét, így sok frekvencia felhasználása szükséges egy ilyen rendszer üzemeltetéséhez. A felhasználható frekvenciák száma viszont korlátozott, valamint áthallási okok miatt az azonos frekvenciák egymáshoz közel fekvő adóknál nem alkalmazhatók.

Ezért olyan célszerű frekvenciakiosztást kell javasolni, amely a frekvenciák ismételt felhasználásával csökkenti a szükséges frekvenciák számát és a „lóugrásos" latin négyzeteken alapuló frekvenciakiosztással megakadályozza mind az áthallást, mind a zavarást. A megoldást a cikk első részében említett Nasik-négyzet alkalmazása biztosítja.

A *8. ábrán* bemutatott Nasik-négyzet (ez egy olyan Nasik-négyzet amely egyben latin négyzet is és rendelkezik a lóugrás-tulajdonsággal) olyan tulajdonságú, hogy minden egyes eleme (egy elem egy frekvencia kiosztást reprezentál) nyolc olyan elemmel szomszédos, melyek közül két szomszédos elemhez nem rendelhető azonos frekvencia. Az ábrán bemutatott latin négyzet ötödrendű, ez azt jelenti, hogy a javasolt eljárás használatához legalább öt különböző frekvencia szükséges. Természetesen az egy adó-vevőhöz rendelt frekvenciák száma egynél jóval több is lehet.

0	1	2	3	4
2	3	4	0	1
4	0	1	2	3
1	2	3	4	0
3	4	0	1	2

8. ábra

Egy frekvencia-ugratásos hírközlő rendszer úgy jellemezhető, hogy *n* frekvenciát használhat a rendszerben működő legfeljebb *n* adó. Az adók mindegyike bizonyos időközönként frekvenciát vált a zavarás megnehezítése végett vagy egyéb okokból. A frekvenciaugratásos hírközlő rendszer akkor hatékony, ha a rendszerben működő adók egymástól függetlenül (külső szinkronizálás nélkül) működhetnek azonos frekvenciákat használhatnak oly módon, hogy az ütközés (két adónak egy időben azonos frekvencia használata) elkerülhető legyen.

R. D. Yates és G. R. Cooper már 1966-ban készített kutatási jelentésükben javasolták latin négyzetek alkalmazását a frekvencia-ugratás hatékonyságának növelésére.

4. Alkalmazás a digitális képkódolás és átvitel területén

Egy (0,1) mátrixról akkor mondjuk, hogy *uxv* (u,v ≥ 2) *horizontális ablak tulajdonsággal* rendelkezik, ha egy *u* sorból és *v* oszlopból álló ablakot horizontálisan mozgatva a mátrixon, minden nem csupa nullából álló ablak legfeljebb egyszer fordul elő. (Hasonlóképpen definiáljuk a vertikális ablak tulajdonságot.)

Egy mátrixot akkor mondunk *uxv* ablak tulajdonságúnak, ha horizontálisan és vertikálisan is *uxv* ablak tulajdonságú.

Természetes általánosítása a fogalomnak, ha egy latin négyzettől követeljük meg a horizontális, illetve vertikális ablak tulajdonságot. (0,1) mátrixokra az ablak tulajdonságot két Bell laboratóriumban dolgozó matematikus (F. J. MacWilliams, N. J. A. Sloane vizsgálták először. Az ablak tulajdonsággal rendelkező mátrixok szerkesztésének gyakorlati alkalmazása is van, például a digitális képkódolás és átvitel területén.

Most bemutatunk egy példát keresztül egy ablak tulajdonságokkal rendelkező mátrix szerkesztést, amely – mint látni fogjuk – a teljes latin négyzeteken alapul.

Tekintsünk egy negyedrendű teljes $L_4(a_{ij})$ latin négyzetet *(9. ábra)*, majd utolsó oszlopát megismételve, valamint egy kizárólag ötösöket tartalmazó oszlopot hozzávéve kapjuk a *10. ábrán* látható $M_{4x6}(b_{ij})$ mátrixot.

$$L_4(a_{ij}) = \begin{matrix} 4 & 1 & 3 & 2 \\ 1 & 2 & 4 & 3 \\ 2 & 3 & 1 & 4 \\ 3 & 4 & 2 & 1 \end{matrix}$$

9. ábra

$$M_{4x6}(b_{ij}) = \begin{matrix} 4 & 1 & 3 & 2 & 2 & 5 \\ 1 & 2 & 4 & 3 & 3 & 5 \\ 2 & 3 & 1 & 4 & 4 & 5 \\ 3 & 4 & 2 & 1 & 1 & 5 \end{matrix}$$

10. ábra

A teljes latin négyzet tulajdonságból következik, hogy úgy a 9. ábra $L_4(a_{ij})$ latin négyzete, mint a 10. ábra $M_{4x6}(b_{ij})$ kiterjesztése rendelkezik mind vertikális, mind horizontális *2x2* ablak tulajdonsággal.

Vesszőmentesnek nevezünk egy C kódot, amely *n* hosszúságú szavakból áll, ha bármely $a_1...a_n \in C$ és $b_1...b_n \in C$ esetén az $a_j a_{j+1}...a_n b_1 b_2...b_{j-1}$ (j=2,3,...,n) kódszavak egyike sincsen C-ben.

Szemléltető példaként a 10. ábra mátrixát és a *11. ábra* bináris vesszőmentes kódját felhasználva, az *i, j* $\leftrightarrow c_{b_{ij}}$ *(i=1,2,3,4 j=1,2,3,4,5,6)* megfeleltetéssel nyerjük a *12. ábrán* látható (0,1) mátrixot, amely *14x1* ablak tulajdonságú.

$c_1 = 00001$ $c_2 = 01100$ $c_3 = 01001$ $c_4 = 01110$ $c_5 = 01101$ $c_6 = 01111$

11. ábra

		j																													
			1				2				3				4				5				6									
	1	0	1	1	1	0	0	0	0	0	1	0	1	0	0	1	0	1	1	0	0	0	1	1	0	0	0	1	1	0	1
i	2	0	0	0	0	1	0	1	1	0	0	0	1	1	1	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	1
	3	0	1	1	0	0	0	1	0	0	1	0	0	0	0	1	0	1	1	1	0	0	1	1	1	0	0	1	1	0	1
	4	0	1	0	0	1	0	1	1	1	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0	1

12. ábra

A fenti latin négyzeteket felhasználó szerkesztési mód részletes leírása megtalálható [3]-ban.

5. Latin négyzet alapú párhuzamos aritmetika

A számítástechnika egyik égető problémája a műveletvégzési idők lerövidítése. Ennek érdekében jöttek létre a különböző párhuzamos műveletet végző architektúrák (pipeline processzorok, array processzorok stb.)

Nem kaptak eddig megfelelő szerepet az úgynevezett maradék számrendszerbeli ábrázoláson alapuló aritmetikai egységek. Ilyen aritmetikai egység ismertetésére és a latin négyzetekkel való kapcsolatára derül fény a következőkben. Ezen példa kapcsán rámutatunk a latin négyzetek művelet táblaként való kódolási alkalmazására.

A számok maradék számrendszerbeli ábrázolásának kezdetei az ókori Kínában keletkezett, úgynevezett *kínai maradéktételen* alapszik. Mivel e tétel több mint kétezer éves, nem természetes, hogy a soros műveletvégzés lett a számítástechnika születésénél az első és napjainkig is alapvetően elterjedt architektúra.

Néhány olyan tudomány és technikatörténeti tényt igyekeztem [4] könyvemben közkincsé tenni, amelyek ezt a jelenséget még érdekesebbé teszik. Eme kevéssé ismert tények szerint már a XX. század első felében történt kísérlet a maradék számrendszerbeli ábrázoláson alapuló célgép (prímszám szita) készítésére, később D. N. Lehmer, majd fia D. H. Lehmer hasonló célú és megoldású fotoelektronikus gépet épített. Az ifjabb Lehmer, aki ott volt az ENIAC születésénél, annak soros architektúrájúvá tételéről a következőket írta:

„A következő dátumunk 1946, ami természetesen az ENIAC éve. Vajon felhasználható-e a nagysebességű számítógép a szita-módszer elvégzésére? Ez egy magas párhuzamosságú gép volt, amíg von Neumann el nem rontotta."

A kínai maradék tétel:

Pontosan egy olyan $x<n$ természetes szám van melyre fennállnak a következő kongruenciák, $x \equiv a_1 \bmod m_1$, $x \equiv a_2 \bmod m_2$,...., $x \equiv a_k \bmod m_k$, ha $m_1, m_2, ..., m_k$ páronként relatív prímek és $n = m_1 \cdot m_2 \cdot ... \cdot m_k$.

Az $x \equiv a_i \ mod \ m_i$ írásmód azt jelenti, hogy az x számot m_i-vel osztva a_i maradékot ad. Azt mondjuk, hogy az x szám maradék számrendszerbeli ábrázolása az $(a_1a_2....a_k)$ vektor.

Az összeadás, kivonás, szorzás a maradék számrendszerben ábrázolt számok között komponensenként történik. Legyen x maradék számrendszerbeli ábrázolása $(a_1a_2....a_k)$ és y ábrázolása $(b_1b_2....b_k)$, akkor $x+y$ ábrázolása $(a_1 \oplus b_1 a_2 \oplus b_2 ... a_k \oplus b_k)$, ahol a $\oplus$ művelet a $mod \ n_i \ (i = 1,2,...,k)$ összeadást jelenti. Hasonló módon értelmezhető a kivonás és a szorzás is.

A *13. ábra* bemutatja a 0-15 közé eső egész számok 3 és 5 modulusra vonatkozó maradék számrendszerbeli alakjait, majd egy példán keresztül érzékeltetem a maradék számrendszerben való összeadást. Eme táblázat segítségével már képes bárki ebbe a szám intervallumba eső számokkal műveleteket végezni, ezáltal megtapasztalni azt a meghökkentő lehetőséget, amit a párhuzamos számítás (párhuzamos aritmetika) jelent.

13. ábra
Táblázat a 0-29 természetes számok maradék számrendszerbeli ábrázolására

Egész számok	m_1 3	m_2 5
0	0	0
1	1	1
2	2	2
3	0	3
4	1	4
5	2	0
6	0	1
7	1	2
8	2	3
9	0	4
10	1	0
11	2	1
12	0	2
13	1	3
14	2	4
15	0	0

A maradék számrendszerben az összeadás, vagy a kivonás sokkal gyorsabban, valóban az összes számjegyen szinte egyszerre (párhuzamosan) elvégezhető, mivel nincs átvitel. Illusztrációként a *14. ábrán* bemutatjuk a következő műveletek elvégzését modulus rendszerben: *7+3+1=11.*

Össze- Adandók	m1 3	m2 5
7	1	2
3	0	3
1	1	1
Eredmény:	**2**	**1**

14. ábra

Az összeadás eredménye *(2,1)* a 13. ábra 11. sorában megtalálható, vagyis a modulus alakban kapott eredmény valóban a tízes számrendszerben kiszámított *11-nek* felel meg.

A kínai maradék tétel tehát ad egy olyan szám ábrázolási módot, amelynek segítségével a párhuzamos műveletvégzés könnyen megvalósítható.

A modulus aritmetika műveleteihez tartozó művelet táblák egy-egy latin négyzetet alkotnak. A fent bemutatott *mod 3*, illetve *mod 5* összeadáshoz tartozó művelet táblákat mutatják a *15. ábrán* látható peremezett latin négyzetek:

$\oplus_3$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

$\oplus_5$	0	1	2	3	4
0	0	1	2	3	4
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

15. ábra

Most példát adunk a *mod 3* és *mod 5* modulus aritmetikabeli szorzásra, illetve a szorzás elvégzéséhez szükséges művelettáblákra, amelyek a perem sor, illetve oszlop elhagyásával latin négyzetet alkotnak *(16. ábra).*

$\oplus_3$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

$\oplus_5$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	2	1	4	2
4	0	4	3	2	1

16. ábra

Az összeadásnál bemutatott 13. ábra szerint:

$3=(0,3), \ 5=(2,0),$ ekkor $3 \cdot 5=(0 \oplus_3 2, \ 3 \oplus_5 0)=(0,0)$,

amely a táblázat 15. sorában található, tehát a szorzás eredménye *15*, ami megfelel a tízes számrendszerbelinek.

A bemutatott példákból is kiderül, hogy a modulus aritmetika használatának fő nehézségét a túlcsordulás, valamint az előjel jelzése és a maradék számrendszerbeli ábrázolásból a tízes számrendszerbe és viszont konvertálás viszonylagos lassúsága jelenti. Ezt az ábrázolás módot tehát ott célszerű alkalmazni, ahol a számolás igény nagy és csak ritkán kell konvertálni a számokat.

Irodalom

[1] A. M. Andrew: Decimal error-correction: a solution, Computer J. 18 (1975), pp.284–285.

[2] J. Dénes: Latin squares and non-binary encoding, Proc. Conf. Inform. Theory (Cachen, France, 1977), CNRS, Paris 1979., pp.215–221.

[3] J. Dénes, A. D. Keedwell: A new construction of twodimensional araye with window property, IEEE Trans., On Information Theory 1988.

[4] Dénes Tamás: Titkos-számítógép-történet, Aranykönyv Kiadó, Budapest, 2003.

[5] A. Ecker, G. Poch: Check character systems, Computing 37(1986), pp.289–297.

[6] D. R. Hughes, F. C. Piper: Design theory, Cambridge University Press, 1985.

[7] D. H. Lehmer: A photo-electric number sieve, Amer. Math. Monthly 40 (1933), pp.401–406.

[8] D.H.Lehmer: A machine for combining sets of linear conqruences, Mathematische Annalen 109 (1934), pp.661–667.

[9] R. Mandl: Orthogonal latin squares: an application of experimental design to compiler testing, Comm. ACM 28(1985), pp.1054–1058.