tanulmányok 89/1979

MTA Számítástechnikai és Automatizálási Kutató Intézet Budapest

MAGYAR TUDOMÁNYOS AKADÉMIA
SZÁMITÁSTECHNIKAI ÉS AUTOMATIZÁLÁSI KUTATÓ INTÉZETE

A SIS77 STATISZTIKAI INFORMÁCIÓS RENDSZER

(A FELHASZNÁLT SZÁMITÁSTECHNIKAI ESZKÖZÖK, A RENDSZER SZERKEZETE ÉS PROGRAMJAI)

Irta:

RUDA MIHÁLY

Tanulmányok 89/1979.

A kiadásért felelős:
DR VÁMOS TIBOR

ISBN 963 311 067 X

ISSN 0324-2951

Készült a
KSH Nemzetközi Számítástechnikai Oktató és Tájékoztató
Központ Reprográfiai Üzemében

7910163 MTA KESZ Sokszorosító. F. v.: dr. Héczey Lászlóné

Tartalomjegyzék

Bevezetés

A SIS77 /Statistical Information System 1977/ egy általános célú statisztikai adatfeldolgozó rendszer. Elkészitését az országos kórházi morbiditási vizsgálatok statisztikai feldolgozása tette szükségessé /ld. [1], [13] /. A rendszer felépitése azonban lehetővé teszi, hogy más statisztikai felmérésnél is használhassuk. A rendszer kialakulásának körülményeivel és a felhasznált módszerekkel, elvekkel a [2] és [13] tanulmány foglalkozik részletesebben. Most a rendszer szerkezetét, az egyes programok felépitését, müködését és a felhasználás technikai kérdéseit tárgyaljuk.

Ismételten ki kell emelni azt, hogy bár rendszerünk a kórházi morbiditási adatok feldolgozásának érdekében készült, és ennek az információs rendszernek az igényeit maximális mértékben igyekeztünk kiszolgálni, mégis a létrehozott számitógépes rendszer minden részében általános fogalmakon nyugszik, és általános adatfeldolgozási feladatokat old meg. Ez a jelenlegi kórházi morbiditási rendszerben is szükségszerü volt /[1], [8], [9], [13] /. Az alapadatokban, a szolgáltatott információkban bekövetkező változások ugyanis nem teszik lehetővé egy minden részében

rögzitett rendszer alkalmazását. Ugyanakkor egy általánosan használható rendszer nyilván értékesebb is. Itt kell megemliteni, hogy rendszerünk továbbfejleszthető, kiterjeszthető és más rendszerekhez is kapcsolható. Ilyen irányú javaslatainkat, elképzeléseinket a [13] tanulmányban fogalmaztuk meg.

A jelen tanulmány első részében egy áttekintő leirást adunk, amelyből a felhasználó képet kaphat a rendszer céljáról, müködési módjáról, a felhasználási lehetőségekről.

A következő részben szerkezeti, számitástechnikai oldalról /és nem a felhasználó oldaláról/ mutatjuk be a rendszert. Foglalkozunk a felépités elveivel, a programok és adatfile-ok kapcsolatával, a feldolgozást összefogó adminisztrációs részrendszerrel és a rendszer betöltésével.

A harmadik rész az egyes eljárások részletes leirását tartalmazza. Az eljárások felhasználási módját és a programleirásokat külön dokumentációkban adjuk meg, mivel ezek már olyan részleteket tartalmaznak, amelyek nem tartoznak a rendszer lényegéhez, és egyes részletekben változhatnak is. Ugyanigy a rendszer betöltésének részletes, és a pillanatnyi állapotnak megfelelő leirását is külön dokumentációban célszerü megadni.

A rendszer kialakitásának munkáiban - matematikai, statisztikai, számitástechnikai, szervezői, programozói feladatok elvégzésében - az MTA SZTAKI Valószinüségszámitási és Matematikai Statisztikai Osztályán GÁL ANNA, GARÁDI JÁNOS, KOVÁCS KÁLMÁN, KRÁMLI ANDRÁS, RATKÓ ISTVÁN, RUDA MIHÁLY, SOLTÉSZ JÁNOS, SZILLÉRY ANDRÁS és VASS RÓZSA vett részt. A rendszert a kórházi morbiditási vizsgálatokban az ESZTIK munkatársai sikeresen alkalmazták /[1], [13]/.

A szerző köszönetet mond SZMRECSÁNYI KLÁRÁNAK a rendszer Honeywell gépen történt megvalósitásakor adott hasznos tanácsaiért.

A statisztikai táblázatokat készitő részrendszer kialakitásában sikeresen alkalmaztunk egy NYIRY GÉZA által javasolt módszert /ld. 3.8. pont/, amelyet a szerző egy 1968-ban készült kisebb méretü statisztikai rendszerben már felhasznált - ld. [3].

1. A rendszer általános leirása

1.1 Egy statisztikai adatfeldolgozási folyamat

Ebben a tanulmányban a statisztikai adatfeldolgozás néhány számitástechnikai kérdését tárgyaljuk. Igy nem kerül sor az adatfelvételek előkészitői, szervezői munkáinak leirására - ezzel részben a [2] tanulmány foglalkozik. Nem foglalkozunk a feldolgozásokat előkészitő döntések, előzetes felmérések kérdésével, vagy a nyert információk felhasználhatóságának problémájával. Ezek más - pl. az orvosi - szakterület feladatai. Abból a fázisból indulunk tehát ki, amikor már rendelkezésünkre áll egy statisztikai minta. Ebből a mintából kiván a felhasználó információkat - statisztikai táblázatokat, mutatókat - képezni.

Bármely adatfeldolgozó rendszerben elsőként ellenőrzési, átkódolási, javitási müveleteket kell elvégezni.

A hibátlan és megfelelő módon kiegészitett adatrendszeren a vizsgálatokat sokszor a rendelkezésre álló adathalmaznak csak egy részén kell folytatni, ezért szelektálni kell az adatokat.

Statisztikai feldolgozásoknál általában szekvenciális file-okat célszerü használni. Bizonyos esetekben azonban bonyolultabb adatbázisokhoz kell kapcsolódni. Ilyenkor a hatékony feldolgozást olyan programok biztositathatják, amelyek egységes formában elkészitik a statisztikai feldolgozás által használt file-okat /szekvenciális file-ok/.

Statisztikák készitése más adatfeldolgozási feladatoktól többek között abban tér el, hogy nem egyedi adatokkal, hanem egyedek tipusaival dolgozunk, és a statisztikai feldolgozások ebben a vonatkozásban lényegesen különböznek az "adatnyilvántartó" rendszerektől /[9], [13]/. A statisztikai adatok létrehozásakor - legyenek ezek egyszerü gyakoriságok vagy a bonyolultabb matematikai statisztikai analizisekhez szükséges összegek, szorzatösszegek, négyzetösszegek - célszerü már a tipusokból és nem az egyedekből kiindulni /[13]/.

A statisztikai adatszolgáltatás befejező - és a felhasználó oldaláról nézve egyik leglényegesebb - része az adatok kinyomtatása, képernyőn való megjelenitése, stb. Ez táblázatok, grafikonok készitését és statisztikai analizisek elvégzését jelenti.

Bonyolult rendszereknél mindenképpen szükséges egy a rendszer müködését összefogó, és a feldolgozásokhoz szükséges segédadatokat kezelő adminisztrációs segédrendszert létrehozni.

A következő pontban a kórházi morbiditási adatfeldolgozás példáján bemutatjuk, hogy az itt vázolt folyamat gyakorlati megvalósitása milyen formában történhet.

1.2 Egy példa statisztikai adatfeldolgozásra

Most a kórházi morbiditási vizsgálat esetére sorravesszük az előző pontban felsorolt feldolgozási fázisokat.

Nyilvánvaló, hogy az adatszolgáltatóktól beérkező és gépi feldolgozásra alkalmas formában rögzitett adatokat ellenőrizni kell. Ez a kórházi morbiditási vizsgálat esetében az egyes adatelemek értékeinek vizsgálatán túl bonyolultabb összefüggések ellenőrzését is jelenti. Ezek az összefüggések időnként változhatnak, ezért az ellenőrzési feltételek megadását egy rugalmasan változtatható rendszerben kellett megoldani /[1]/.

Néhány példa a szóbanforgó feladatok közül: ellenőrizni kell például, hogy az adatok között a pillanatnyilag létező kórházak kódjai szerepeljenek /egyes intézetek megszünnek, összevonások történnek, új kórházak épülnek/ és az egyes kórházakban csak az ott

müködő osztályok kódjával kerüljenek feldolgozásra adatok. Ellenőrizni kell, hogy néhány kiemelt betegség kivételével ne szerepeljen túl hosszú ápolási idő, és a diagnózisok a beteg nemével, korával összeférhetők legyenek. Ezek az ellenőrzések sok esetben helyettesithetik az egyébként szükséges - és csak magas végzettségü munkaerővel elvégezhető - szakmai ellenőrzést. Ha figyelembe vesszük, hogy pl. a kórházkódok jelenleg egy közel 2000 hosszuságú intervallumon, a diagnóziskódok egy 10000 hosszuságú intervallumon változhatnak, akkor elmondhatjuk, hogy az itt emlitett ellenőrzési problémák egy bonyolult számitástechnikai feladatot jelentenek.

Az ellenőrzési feladatok mellett új adatokkal is ki kell egésziteni az alapadatrendszert. Ezt a munkát is célszerü a feldolgozás első fázisában, az ellenőrzéssel együtt elvégezni. Ilyen feladat például a négyjegyü diagnóziskód röviditett jegyzékekre való leképezése, vagy a születési adatok alapján az életkor kiszámitása, stb.

Az ellenőrzéshez sorolható a minta reprezentativitásának vizsgálata is. Ez a megfelelő adatok szerinti egy vagy többdimenziós gyakoriságeloszlások meghatározásával történhet. Például a jelenlegi kórházi morbiditási vizsgálatban a minta szakmánként pontosan 10 %-os kell hogy legyen.

Szelekciós feladatokra a következő példák adhatók. Ki kell választani egy bővebb mintából a fent emlitett 10 %-os mintát. Legtöbb esetben csak a magyar állampolgárságú betegek kórházi ellátását kell vizsgálni. Előfordul, hogy egy-egy kórház vagy megye a saját adataiból készült statisztikákat igényel. Speciális orvosszakmai vizsgálatok esetén az előforduló diagnózisok egy részét kell csak feldolgozni, például a nőgyógyászati diagnózisokon belül a szülészeti események különválasztása indokolt.

Bonyolultabbnak mondható adatbázisszervezési probléma akkor merül fel, amikor nem az adatfelvétel alapegységét képező ápolási esetre kivánunk statisztikákat készitени, hanem például az eseteket személyenként összevonjuk, és igy a természetes személyekre vonatkozó statisztikákat készitünk /pl. egy személy többszöri ápolása is igy vizsgálható/. Hasonló problémát kell megoldani az ápolási esetenként változó számban előforduló kisérő illetve következményes betegségek vizsgálatakor is.

A statisztikákban szereplő tipusok létrehozásával kapcsolatban a következők mondhatók el. A kórházi morbiditási vizsgálatban szerzett tapasztalataink szerint /de valószinü, hogy más információs rendszerekben is igy van/ a felhasználók számára lényegesen különböző

információt adnak formailag lényegtelen részletekben eltérő statisztikai táblázatok is. Például adott változók /mondjuk életkor, nem, foglalkozás/ szerint bontott táblázatban más és más szempont szerint készithetünk százalékos megoszlást, egyes bontási szempontokat elhagyhatunk vagy összevontabb formában közlünk, stb. Ezért célszerü az információszolgáltatásba egy közbeeső fázist iktatni, amelyben a tipusok - a kivánt szempontok szerinti osztályozás - egy részletesebb bontását hozzuk létre, amiből később többféle összevonás és egyéb formai változtatás /kumulálás, részösszegképzés, görbekirajzolás, stb./ útján sokféle táblázatot készithetünk.

A táblázó részrendszer feladatát az előbb emlitett adatmanipulációkon /összevonás, kumulálás, grafikonhoz használt értéktáblázatok kitöltése/ túl a megjelenités formai megoldása is képezi. Változtatható fejléceket kell kiirni, adatelnevezéseket kell a táblázatokba illeszteni. Ezen kivül kiegészitő adatokat - arányszámokat, átlagokat, százalékokat - is létre kell hozni, és a grafikus ábrázolást is meg kell oldani.

A kórházi morbiditási vizsgálatban a leggyakrabban a következő adatok kerülnek a táblázatokba: ápolási eset és nap, ezek százalékos megoszlása, átlagos ápolási nap, tizezer lakosra jutó eset és nap, grafikon

készitése a tizezer lakosra jutó eset és nap értékeiről.

Természetesen más értékek is szerepelhetnek a táblázatokban, például a tizezer lakosra jutó értékek helyett egyrészt kisebb vagy nagyobb populációegységhez is hasonlithatunk /pl. 1000 lakosra jutó értékek/, másrészt más jellegü adatok, például egy kórházi ágyra jutó eset és nap is szerepelhetnek. Ugyanigy az ápolási napok összege helyett a mütött betegeknél a mütétig eltelt napok számának összegét, vagy akár az adott táblapoziciónak megfelelő részpopulációban bekövetkező halálozások számát is megadhatjuk. Ilyenkor az átlagos ápolási nap helyett a mütétig átlagosan eltelt idő illetve az egy főre jutó halálozási arányszám szerepel.

Végezetül a rendszeradminisztráció fontossága a következőkkel jellemezhető: A jelenlegi kórházi morbiditási vizsgálatban közel 100-féle táblázat készitését igényelték a különböző szintü felhasználók. Ez a feladat az előkészitő lépésekkel együtt adatfile-ok nagy tömegének kezelését teszi szükségessé. Az itt mutatkozó nehézségek még fokozódhatnak, ha például egy időben több felmérés anyagát dolgozzuk fel. Az adatfile-ok mellett tárolni és kezelni kell a táblázatokban felhasznált nagytömegü elnevezést /diagnózisnevek, kórháznevek, stb./ és statisztikai adatot /pl. népességstatisztikai adatok a tizezer lakosra jutó értékek számitásához/ is.

1.3 A SIS77 felhasználási jellemzői

Ebben a szakaszban nagyvonalakban bemutatjuk azt, hogy a felhasználó milyen módon kezelheti a SIS77 rendszert és milyen lehetőségei vannak a rendszer alkalmazásában.

A SIS77 önálló egységekből - modulokból - áll. Ezek a modulok a feldolgozás egy-egy önálló részfeladatát végzik el /ezeket a feladatokat soroltuk fel az 1.1 pontban/. Egy modul ismételten is felhasználható, például egy ellenőrzött és kiegészitett adatállományt újabb ellenőrzéseknek és kiegészitő eljárásoknak vethetünk alá.

A modulok egy vagy több rendszerprogramból állnak. /A modulok illetve a rendszerprogramok belső szerkezetét a következőkben tárgyaljuk./ A rendszerprogramok FORTRAN nyelven készültek - a rendszerépités, a rendszermüködés és a felhasználás szempontjait figyelembevéve ezt a nyelvet volt célszerü használni. A SIS77-ben egyéb software eszközt - a szokásos rendezési eljáráson kivül - nem alkalmaztunk. A rendszer igy egységes és más gépekre is átvihető /jelenleg egy Honeywell 66/60-as gépen müködik/. Ugyancsak az általánosan használt FORTRAN nyelv alkalmazása teszi lehetővé, hogy a rendszer egyes moduljait más rendszer-

ben is felhasználhassuk. A kapcsolódási módokkal és azok kiterjesztésének lehetőségeivel a [13] tanulmányban foglalkozunk.

A rendszer részei /az 1.1 pontnak megfelelően/: ellenőrzés és kiegészités, szelekció, adatszervezési eljárások /ld. az 1.2 pontban a természetes személyek és a kisérőbetegségek problémakörét/, statisztikai adatok /tipusok/ előállitása, az adatok megjelenitése /táblázás/, adminisztráció. A szelekciós feladatokat ellátó részben /az adattárolás optimalizálásának érdekében/ a rendszer konvertálja az eredeti /karakter formában tárolt/ adatállományt.

A rendszermodulok - az egyszerübb feladatok kivételével - nem előre elkészitett programokból állnak. A feldolgozásokat végző programokat a felhasználó által megadott feladatleirás alapján a rendszer maga generálja. Ezzel mind a rendszermüködés mind a felhasználás hatékonyabb lesz /ld. [10], [11], [12]/. A rendszerprogramok ugyanis mindig az aktuális feladatnak megfelelő formában készülnek el, igy gépidőben és tárkapacitásban kisebb igényüek mint az általános formában megfogalmazott programok, és szerkezetük is egyszerübb /könnyebbé téve ezzel a rendszer karbantartását és továbbfejlesztését/. A felhasználónak viszont csak ugyanannyi információt kell közölnie

a rendszerrel mint a hagyományos programozási technika esetén /az elvégzendő feladatokat mindenképpen meg kell határozni/, de a programok alkalmanként történő generálása nagyfokú rugalmasságot és igy kényelmes felhasználást is biztosit. /Feladatonként rögzitett, paraméterekkel nem vezérelhető programok alkalmazása szóba sem jöhet, hiszen a rendszer alapját képező kórházi morbiditási vizsgálatban az alapadatok, az ellenőrzési feltételek, az információigények változása miatt a rendszerprogramokat állandóan módositani kellene, vagy új programok irására lenne szükség - ld. [13]/.

Hogyan irhatja le a felhasználó az elvégezni kivánt feladatokat? A SIS77 egy speciális célnyelv segiségével müködtethető. Ez a célnyelv a statisztikai adatfeldolgozás - az 1.1 pontban emlitett - különböző fázisainak leirására alkalmas. A célnyelv utasitásai a "job control nyelv" utasitásai /vezérkártyái/, paraméterkártyák és egyéb szöveges utasitások.

A job control nyelv utasitásai a szokásos módon biztositják a rendszerprogramok forditását, betöltését, a szükséges adatfile-ok megnyitását. A SIS77 tehát nem egy zárt rendszer, hanem a szokásos módon kezelhető programokból áll. Igy a rendszer feladatait a felhasználó a hagyományos job-felépitésnek megfelelően állithatja össze.

A paraméterkártyák és a szöveges utasitások mindig az egyes rendszerprogramoknak szólnak, vagyis a rendszerprogramok által beolvasott és értelmezett adatok. A felhasználó tehát kiválasztja a feldolgozás következő lépéséhez szükséges rendszerprogramokat. Forditás és betöltés /inditás/ után megadja a szükséges paramétereket és szöveges utasitásokat /és a szükséges input-output file-okat/. A rendszer a feladatleiró utasitásokat /paramétereket/ részletesen megvizsgálja, hibás feladatleirás esetén kiirja a hibát. A programfutások részeredményeinek kiirásával pedig tartalmi ellenőrzésre is módot ad. A paraméterezés és a hibajelzés részletes leirását a 3. pontban illetve külön programdokumentációkban adjuk meg.

Tekintsünk néhány példát! Milyen jellegü utasitások szerepelhetnek a SIS77-ben?

Egyes feladatokat egy-egy paraméterrel meg lehet adni - például igy lehet jelezni azt, hogy a vizsgált adatállomány milyen tipusú ellenőrzését kivánjuk végrehajtani /ld. a 3.1 pontot/, vagy az adminisztrációs részrendszer mely funkcióját kivánjuk aktivizálni /ld. 3.9 pont/.

Egy részfeladat kiválasztása után további paraméterekkel /utasitásokkal/ irhatjuk le az elvégzendő

müveleteket. Uj adatok létrehozása esetén például a következő utasitást adhatjuk meg: /ld. [16]/

4 -12 98-107,54,220-223

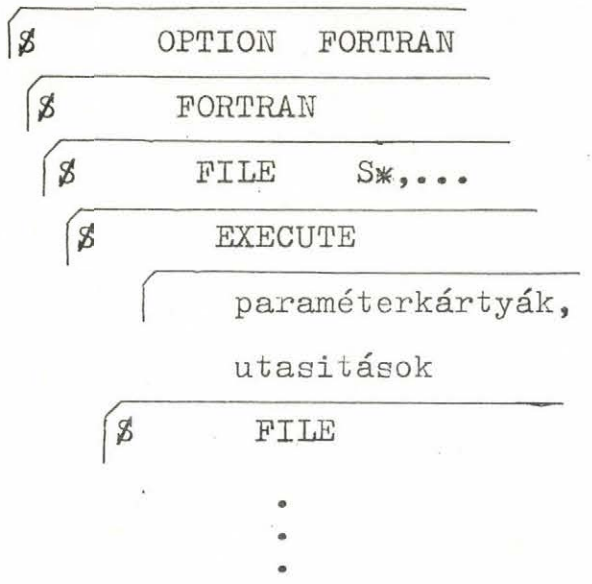
jelentése: az eddig használt kódértékek helyett - ha az előzőkben másként nem intézkedtünk /ezt jelzi a sor elején álló 4-es kód/ - a -12 értéket használjuk, amennyiben az eredeti x kódértékre: $98 \leqq x \leqq 107$ vagy $x=54$ vagy $220 \leqq x \leqq 223$.

Egy másik példa: háromdimenziós táblázatot akarunk készíteni a "KÓRHÁZ", az "OSZTÁLY" és a "NEM" nevü adatokból. Ekkor a

3
KÓRHÁZ OSZTALY NEM

kártyákat adjuk meg. Természetesen egyéb kiegészitő információk is szükségesek: mely file-on találhatók az alapadatok, milyen összevonásokat részösszegeket kivánunk képezni, milyen fejléccel és adatnevekkel látjuk el a táblázatot, stb. Ezeket az információkat is a fentihez hasonló formában adjuk meg.

Egy feldolgozási lépés a következőképpen állitható össze:



fordítás

a soronlévő feladatnak megfelelő program file-ja

betöltés, inditás

input, output file-ok, segédfile-ok

Egy feldolgozási lépés lehet egy programgenerálás, az adatrendszer átalakitása /újabb adatfile létrehozása/, a rendszeradminisztráció módositása /új bejegyzések a rendszeradminisztrációba, régi bejegyzések törlése/, a rendszeradminisztráció lekérdezése, stb. Ennek megfelelően változhat a fenti összeállitás is: például nem FORTRAN programot futtatunk hanem mondjuk a rendszer SORT programját; nem adunk meg output file-t, mert például éppen egy listázó programot futtatunk; egyes szerkesztett programoknál

nincs szükség vezérlő kártyákra, mert a szerkesztő /generáló/ program már minden szükséges információt beépitett a szóbanforgó programba, stb.

A feladatok összeállitásánál ki lehet használni azt, hogy a programgeneráló eljárások vezérlésénél nemcsak az elvégzendő feladat tartalmát, hanem méretét is megadjuk /dönthetünk az ellenőrzési eljárásokban egyidejüleg felhasznált kódtáblázatok méretéről, a kinyomtatott statisztikai táblázatok nagyságáról, stb./, igy mindig alkalmazkodhatunk a pillanatnyilag rendelkezésre álló gépkapacitáshoz is.

Ha például egy feladatot egy lépésben nem tudunk az adott gépen elvégezni, akkor részekre bontva, több lépésben is megoldhatjuk.

2. A SIS77 szerkezete - programok, file-ok, a rendszer betöltése

2.1 A rendszer programjai

A rendszerprogramokat két szempont szerint osztályozhatjuk: egyrészt a müködésmód, másrészt funkció szerint.

Müködésmód alapján elkülönithetők egyszerü programok és generáló-generált programpárok. A programgenerálás optimalizálási célokat szolgál /ld. 1.3 pont/. A SIS77 elsősorban nagyméretü és bonyolult adatrendszerek feldolgozására készült. Ezért indokolt egyes részeiben a feladatok megoldásának optimalizálása [14]. A programgenerálás különböző szinteken történhet: a generált programok összeállitásánál előre elkészitett programrészleteket, szubrutinokat, különálló programsorokat lehet felhasználni - ezeket a rendszer permanens file-okon tárolja. Egyes utasitásokat a generáló programok karakterenként állitanak össze.

Egyszerü programokkal a következő feladatokat oldottuk meg:

gyakoriságtáblázatok készitése közvetlenül az alapadatokon,

a szükséges nagyságú minta kiválasztása,

terminálos táblázóprogram,

adminisztráló program és a rendszer több részében is felhasznált szubrutinok /ld. 3.10 pont/.

Szerkesztő programok generálják:

az ellenőrző és átkódoló programot,

a szelekciókat és a konverziót elvégző programot,

a rekordok összevonását /többször ápoltak, ismétlődő diagnózisok/ és felbontását /kisérőbetegségek/ elvégző programokat,

a közvetlen elérésü file-okat létrehozó eljárást,

a statisztikai adatokat /tipusokat/ létrehozó programot és

a táblázatkiiró programot /batch üzemmódban/.

Funkció alapján elkülönithetők előkészitő jellegü eljárások, az információszolgáltatás fázisa, a rendszeradminisztráció és az általánosan használt szubrutinok.

Előkészitési feladatok:

ellenőrzés, kódolás,
tájékoztató jellegü gyakoriságeloszlások
készitése az alapadatokra,
mintakiválasztás,
egyéb szelekciók,
konverzió,
speciális file-képzési eljárások /rekordok
összevonása és felbontása/,
direkt elérésü file-ok létrehozása
/gyors válogatáshoz/.

Az információszolgáltatás lépései:

statisztikai adatok /tipusok/ létrehozása
- gyakoriságok, kódösszegek,
táblázatok készitése batch üzemmódban és terminálról.

A rendszeradminisztrációt egy olyan program látja el, amely file-ok jegyzékbe vételét, törlését, keresését és segédadatok tárolását teszi lehetővé.

Általánosan használt szubrutinok:

az adatmegfeleltetések /ellenőrzés, kódolás, összevonás/ megadására szolgáló szubrutin /ZSAK/ és

a különböző rendszerprogramoknak az adminisztrációhoz való kapcsolódását segitő szubrutinok - ezek az utóbbiak az egyes rendszerprogramokba beépitett szubrutinok, igy önállóan nem használhatók.

2.2 A rendszer file-tipusai

A file-okat két szempont szerint osztályozzuk: funkciójuk és formájuk szerint.

Funkció szerint megkülönböztetünk:

alapfile-okat - ezeken végezzük az ellenőrzést, mintakiválasztást, új adatok létrehozását és ezekből válogathatjuk ki a további feldolgozásokhoz szükséges részfile-okat,

válogatott, konvertált file-ok - ezekből válogatások, szerkesztési eljárások útján újabb válogatásokat, összevont vagy felbontott rekordokat, rendezett file-okat és direkt elérésü file-okat hozhatunk létre, és ezekből a file-okból nyerhetők a statisztikai adatok /tipusok/ is,

összevont vagy felbontott rekordok file-jai - ugyanúgy használhatók, mint a többi válogatott, konvertált file,

rendezett /válogatott, konvertált/ file-ok - részenkénti feldolgozások céljára,

direkt elérésü, fej rekorddal ellátott file - gyors válogatás céljára,

"táblafile-ok" - a statisztikai adatokat /tipusok gyakoriságait, kódösszegeit/ tartalmazó file-ok,

segédfile-ok - elnevezések, statisztikai adatok,

adminisztráló file,

programfile-ok - a rendszerprogramok tárolóhelye /lehet kártya is/,

a szerkesztő /generáló/ programok által használt file-ok - itt a generált programok összeállitásához szükséges programrészletek vannak.

Forma szerint a következő file-tipusokat különböztethetjük meg:

karakter formájú file-ok /alapfile-ok, segédfile-ok, adminisztrációs file, programfile-ok, programrészletek/,

tömöritett bináris file-ok /válogatott, konvertált file-ok, beleértve az összevont és felbontott rekordok file-jait és a rendezett file-okat/,

direkt elérésü, fej-rekorddal ellátott /tömöritett bináris/ file - a fej rekord mutatókat tartalmaz,

összetett bináris forma - a táblafile /statisztikai adatok file-ja/ bináris formában, egy szóban tartalmazza a gyakoriság és kódösszeg értékeket.

A rendszer belső tárolás céljára használja a tömöritett bináris formát. Egy adat számára csak annyi bitet foglal le a rendszer, ahány bit az illető adat maximális értékének tárolására elegendő. Ilyen módon azonban csak nem negativ egész értékek tárolhatók /ez adatfeldolgozási feladatoknál nem túl szigorú megkötés/. A táblafile-on a gyakoriság és a kódösszeg egy szóban való tárolásával /gyakoriság 15 bit, kódösszeg 20 bit/ ezek felső korlátja rendre 32767 illetve 1048575. A táblafile-on tehát csak olyan finomságú tipusfelbontás szerepelhet, amelyben e két érték egyikét sem lépjük túl.

Az előkészitő fázisban - mivel elsősorban itt jöhet szóba más rendszerekkel való kommunikáció - a szokásos karakteres tárolásmód alkalmazása volt célszerü.

2.3 File-ok és programok kapcsolata

A következő ábrán szemléltetjük, hogy az egyes eljárások hogyan kapcsolódnak egymáshoz a különböző file-tipusokon keresztül.

Az ábrán szereplő file-jelzőszámok jelentése:

1. és 2. az ábra szövege szerint,

3., 4. és 6. tömöritett bináris formában tárolt adatok: 3. egyszerü szekvenciális file, 4. rendezett file, 6. összevont vagy felbontott rekordok file-ja - ez utóbbi lehet rendezett is ,

5. direkt elérésü, fej-rekorddal ellátott file, gyors szelekcióhoz,

7. táblafile /statisztikai adatok file-ja/,

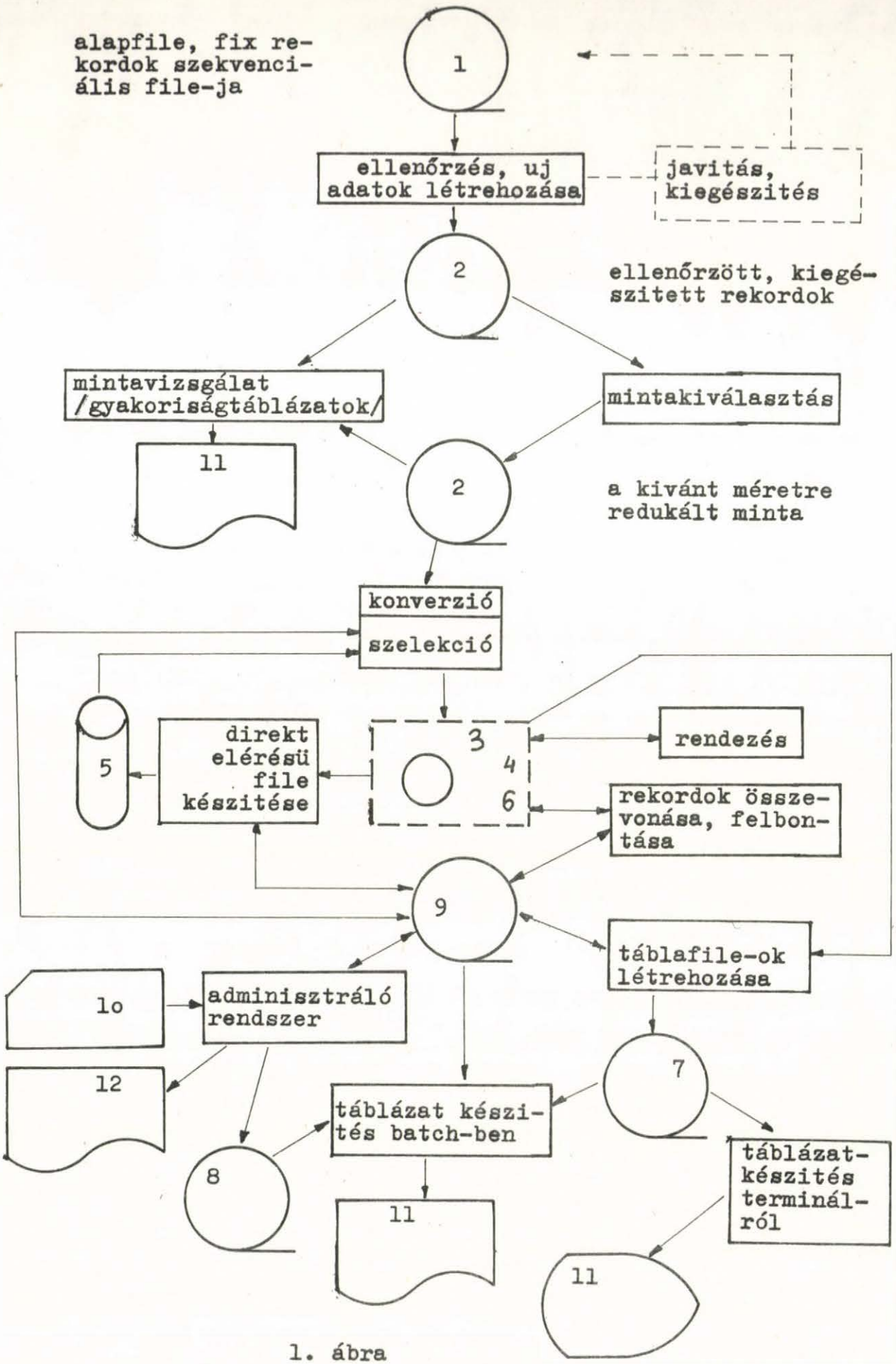
8. segédfile-ok,

9. adminisztrációs file,

10. segédfile-ok bevitele,

11. táblázatok, grafikonok,

12. file leirások.


alapfile, fix rekordok szekvenciális file-ja
1
ellenőrzés, uj adatok létrehozása
javitás, kiegészités
2
ellenőrzött, kiegészitett rekordok
mintavizsgálat /gyakoriságtáblázatok/
mintakiválasztás
11
2
a kivánt méretre redukált minta
konverzió
szelekció
5
direkt elérésü file készitése
3
4
6
rendezés
rekordok összevonása, felbontása
9
táblafile-ok létrehozása
lo
adminisztráló rendszer
12
7
táblázat készités batch-ben
8
11
táblázatkészités terminálról
11

1. ábra

Az ábrán jelzett input-output kapcsolatokon kivül valamennyi program inputjaként vezérkártyák /ezek esetenként segédfile-okon is elhelyezhetők - ld. pl. a ZSAK szubrutin leirását a 3.10 pontban/, outputként pedig sornyomtató /hibajelzések, vezérkártyák kinyomtatása, részeredmények/ is szerepel.

Meg kell jegyezni, hogy - az 5. file kivételével - valamennyi file fix rekordok szekvenciális file-ja.

2.4 A rendszeradminisztráció

Röviden foglalkozunk a rendszeradminisztráció céljával /ld. [13]/. A rendszeradminisztráció az előkészitő fázis után lép be a feldolgozási folyamatba /ld. 1. ábra/. A különböző válogatások, speciális file-képzések /rendezés, rekordok felbontása és összevonása, direkt elérésü file-ok szervezése/, táblafile-ok és segédfile-ok létrehozása folyamán - egy nagyobb feldolgozás esetén - a feldolgozási folyamatban file-ok áttekinthetetlen tömege létezhet egyidejüleg. Egy feldolgozási lépés végrehajtása az input file helyes megválasztásán tul az input és output file pontos leirását is megköveteli. Ezt a feladatot látja el az 1. ábrán látható módon az adminisztráló program. A file-okra vonatkozó információkat egy külön

file-on - az adminisztrációs file-on - helyezi el a rendszer. Ugyancsak az adminisztráló programon keresztül lehet a segédfile-okat a rendszerbe vinni.

2.5 A rendszer betöltése

Ahhoz, hogy a rendszer müködőképes legyen, bizonyos programokat, programrészleteket és egyéb információkat rögzitett elnevezésü permanens file-okon tárolni kell.
Ezek a file-ok a következők:
olyan file-ok amelyeken a szerkesztő /generáló/ programok számára elhelyezett programrészletek vannak; a megfelelő eljárások:

1. ellenőrzés, kódolás,
2. válogatás, konvertálás,
3. rekordok összevonása /természetes személyek, többszörös ápolások/,
4. rekordok felbontása /kisérőbetegségek/,
5. direkt elérésü file-ok szerkesztése,
6. táblafile-ok /statisztikai adatok/ létrehozása,
7. táblakészités /batch üzemmódban/;

az adattranszformációk leirásához szükséges ZSAK szubrutin file-ja.

Ezeket a file-okat a rendszerprogramok maguk nyitják meg, ezért van szükség arra, hogy előre rögzitett elnevezésü file-ok legyenek.

A rendszer müködtetéséhez szükség van az adminisztrációs file allokálására - ez a file a rendszer betöltésekor üres.

Az előbb felsorolt file-ok /a felhasznált Honeywell 66/60-as gépen/ egyenként egy-egy link területet foglalnak el, kivéve a ZSAK szubrutint, amelyhez 2 link lemezterület szükséges. Az adminisztráló file mérete a feldolgozás méretétől, vagyis az adminisztrált file-ok számától függ. Egy file adminisztrálásához 1 littlelink terület kell. /Egy littlelink 320 gépi szó, egy link 12 littlelink./

3. A SIS77 eljárásai

Ebben a fejezetben az egyes eljárások leirását adjuk meg. Elsősorban a felhasznált számitástechnikai eszközöket, az alkalmazott elveket és a felhasználás lehetőségeit mutatjuk be - a kórházi morbiditási vizsgálat példáján keresztül. /A rendszer felhasználását biztositó programdokumentációk (külön) leirásokba kerülnek./

3.1 Ellenőrzés, kódolás

3.1.1 A feladat leirása Ez a rendszermodul az ellenőrzési és kódolási eljárások egységes formában történő megoldását valósitja meg. A felhasznált módszer többek között azon a felismerésen alapszik, hogy az ellenőrzés és az átkódolás formailag ugyanazt a feladatot jelenti, nevezetesen mindkét esetben egy vagy többváltozós függvényeket kell leirni illetve kiszámitani /ld. [13]/. Ellenőrzés esetén a függvényértékek a "jó" és "hibás" értékek, kódolás esetén pedig az új kódértékek.

Ebben a részrendszerben három jól elkülönithető feladatot oldhatunk meg. Az adatok - melyek esetünkben egész értékek /bár ez nem túl erős meg-

kötés, hiszen a számitógépen minden információ végsősoron bináris egész számként ábrázolható/ - kijelölt korlátok közé esését csak a teljesség kedvéért épitettük be. Fejlett technika esetén ez a feladat már az adatrögzités fázisában könnyen megoldható. Nem igy van a bonyolultabb logikai és aritmetikai összefüggések esetén.

Összetettebb ellenőrzési és kódolási feladatoknál két fő irány lehetséges. A hibavizsgálatot illetve a kódolást meghatározó függvényt leirhatjuk a szokásos programozási nyelvek valamelyikén. Ez különösen kényelmes megoldás, ha aritmetikai müveletek elvégzésére és néhány egyszerü feltételvizsgálatra van szükség. Ilyen feladat az, amikor például az alapadatok között egy dátum /év, hó, nap/ van, és ebből kell meghatározni, hogy az illető dátum a hét mely napjához tartozik. Erre a feladatra egy zárt alakban könnyen leirható függvény használható. Más a helyzet akkor, ha a vizsgálni kivánt függvény csak értéktáblázattal adható meg. Ilyenkor az értéktáblázatok egy tömör és jól kezelhető tárolásmódját kell megvalósitani.

A következőkben ezzel a kérdéssel részletesebben foglalkozunk.

3.1.2. Értéktáblázatok leirása és alkalmazása

Rendszerünkben az értéktáblázatok tárolását és feldolgozását a következőképpen oldottuk meg /ld. még [2] , [15] /.

Egyváltozós függvényeknél a teljes értéktáblázatot tároljuk, a szokásos módon: a változó /az operativ tárban lévő/ memóriatáblázat rekeszeinek a cime /relativ cime/, az érték pedig a megfelelő rekesz tartalma. Természetesen ilyen módon csak egész értékeken értelmezett függvényeket tárolhatunk /igaz, hogy adatfeldolgozásban ez a jellemző eset, illetve korlátozott pontossággal adott valós számokat is ábrázolhatunk egész értékként/.

Többváltozós függvényeknél kihasználjuk, hogy a legtöbb esetben az értelmezési tartomány egy olyan többdimenziós mátrix, amelyben a ténylegesen felhasznált változóértékek nagyon ritkán helyezkednek el. Tekintsünk egy példát! Betegségkódok kapcsolatát vizsgáljuk a beteg életkorával és nemével. Legtöbb diagnózis bármely életkorban és mindkét nemnél előfordulhat. Egyes diagnózisok csak a beteg nemével, mások csak az életkorral hozhatók összefüggésbe, megint mások mind az életkorral mind a nemmel kapcsolatban állnak. A kritikus diagnózisok csoportosithatók, pl. nem

mindig szükséges a teljes négyjegyü kód, hanem tizesével összevonva, háromjegyü kódokat is használhatunk.

Az eredeti értelmezési tartomány /változómátrix/ mérete, 10 000-féle diagnózis /0-9999/, 100-féle életkor /0-99/ és kétféle nem esetét számitásba véve:

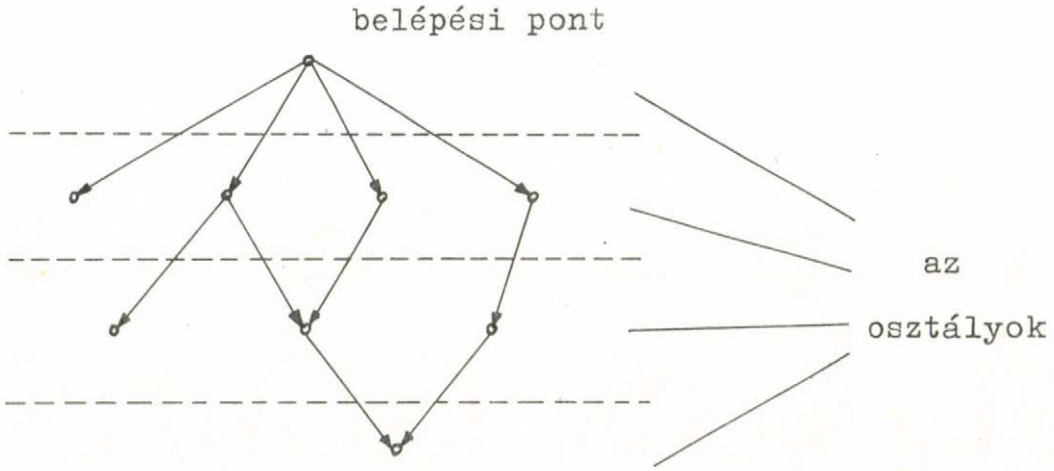
10 000 * 100 * 2 = 2 millió.

Ez egy kezelhetetlen nagyságú értéktáblázat.

Első lépésben - ha lehetséges - összevonásokat végzünk. Az életkor a jelen esetben nem szükséges éves bontásban, elegendő néhány /mondjuk ötféle/ korcsoportot megkülönböztetni /csecsemőkor, gyermekkor, stb./. A másik lehetőség az, hogy - amit egydimenziós esetben is megtehetünk - egyes dimenziókat felbontunk, növelve ezzel a dimenziószámot, de lényegesen csökkentve a tároláshoz szükséges helyet. Példa erre a lehetőségre a fent emlitett diagnóziskód, amelynél a negyedik jegy legtöbb esetben érdektelen, tehát célszerü különválasztani az első három jegytől.

Hogyan tároljuk tehát /az értékösszevonások és dimenziófelbontások után/ a szokásos mátrix-formában adott függvénytáblázatot? A vizsgált függvényt egy speciális irányitott gráfban ábrázoljuk /ld. pl. a 2. ábrát/. Ez a gráf a következő szabályok szerint épül fel.

/1/ A gráf csúcspontjait hierarchikus osztályokba soroljuk. A legmagasabb osztályban egyetlen csúcs lehet. Az irányitás mindig egy eggyel alacsonyabb hierarchiájú osztályban lévő csúcs felé mutathat. Minden csúcsra - kivéve a legfelső osztályban lévő belépési pontot - legalább egy él irányul. /Két csúcsot csak egy él köthet össze./



2. ábra

Erre a gráfra a következő adatfeldolgozási modell épül:

/2/ Az egyes csúcsok táblázatok, amelyek vagy mutatókat /ezek a gráf irányitott éleinek felelnek meg/ vagy függvényértékeket tartalmaznak. A táblázatok rekeszeinek memóriacimei - relativ cimek -

a változóértékek. Minden szinthez egy változót rendelünk. Egy változó több szinthez is tartozhat /igy a szintek száma több is lehet mint a változók száma/. Egy vizsgálat - a függvény kiszámitása - akkor fejeződik be, ha egy olyan rekeszhez érünk, amely nem mutatót, hanem a keresett függvényértéket tartalmazza. /Ez a függvényérték pl. lehet egy hibajelzés is./

Mit is jelent ez a két szabály? A függvény kiszámitásánál mindig az első szintből indulunk ki. Az itt megadott táblázatban megkeressük az első szinthez rendelt változó értékének megfelelő rekeszt /az érték a memóriacim/. Ez a rekesz vagy függvényértéket tartalmaz - és akkor végetér a vizsgálat - vagy egy a következő szinten lévő táblázatra mutat. Ezen a következő szinten a kijelölt táblázatot ugyanúgy vizsgáljuk mint az első szinten lévőt. Itt is vagy befejeződik a vizsgálat, vagy a következő szint egy kijelölt táblázatára lépünk.

Hogyan állitsunk össze egy döntési gráfot? Két szempontot vehetünk figyelembe: Egyrészt minél kevesebb tárolóhelyet kivánunk lefoglalni, másrészt minél rövidebb vizsgálati időt akarunk elérni /vagyis a gráfban a bejárási gyakoriságokat is figyelembe véve az egy irányitás mentén egymáshoz csatlakozó élek számának

átlagát minimalizáljuk/. Az első cél érdekében a nagy terjedelmü táblázatokat olyan szinten helyezzük el, ahol kevés csúcspont, tehát az adott változóra vonatkozó táblázatnak csak kevés példánya van. Az átfutási időt csökkenthetjük, ha a gyakran hivatkozott táblázatokat a felsőbb szintekre helyezzük. Természetesen egy adatleiró táblázat mérete és hivatkozási gyakorisága a döntési gráf szerkezetétől is függ, ezért az optimum megtalálása általában bonyolult feladat.

Tekintsük most az ellenőrzési, kódolási eljárást leiró döntési gráf felépitésének további szabályait!

/3/ Ha egy változóérték - változóérték kombináció - hibás, akkor a döntési gráfban soronlévő táblázatnak a változóértéknek megfelelő rekeszébe -1-et irunk.

/4/ Ha a soronlévő szinten eldönthető, hogy - az esetleges alacsonyabb szintektől /a szintekhez tartozó adatértékektől/ függetlenül - mi a függvényérték, akkor ezt a függvényértéket helyezzük az aktuális táblarekeszbe. A függvényérték - az új kódérték - csak nem negativ egész érték lehet.

/5/ Ha az adott szinten nem lehet dönteni, akkor a megfelelő táblaelem egy mutatót tartalmaz, amely

kijelöli a következő szinten vizsgálandó táblázatot. A mutató csak a következő szintre irányulhat. /Az utolsó szinten nem lehet mutató./ A mutató a döntési folyamatban a soronlévő táblázat sorszáma. A számozás negativ értékekkel történik. Az első táblázatra nem lehet hivatkozni /ld. az /1/ szabályt/. A sorszámozás egy gráfon belül a szintek sorrendjében történik /a mutató abszolut értéke szerint növekvő sorrendben/. A mutató egyesével növekszik. Egy szinten belül a táblázatok sorrendje tetszőleges.

/6/ Egy vizsgálatban több döntési gráf is szerepelhet. Ilyenkor a táblázatok sorszámozása /abszolút értékben/ a gráfok sorrendjében folyamatosan növekszik. A döntési gráfok megadásának sorrendjében egy előző lépésben létrehozott függvényértéket - mint adatot - egy későbbi döntési folyamatban felhasználhatjuk. /Ugyanez igaz az ellenőrzési kódolási eljárás más tipusú feladatleirásaira is - ld. 3.1.1 pont./

/7/ Ugyanahhoz az adathoz tartozó, vagyis a gráf egy szintjén lévő táblázatok értékhatárai /terjedelmük/ nem feltétlenül azonosak. A változóértékek aktuális alsó és felső korlátját ezért minden

táblázatnál külön-külön meg kell adni. A korlátokon kivüleső értékek hibásnak minősülnek. /A korlátok különbözősége abból adódik, hogy egyes adatok a különböző függvénykapcsolatokban nem egyformán minősülnek hibásnak vagy elfogadhatónak./

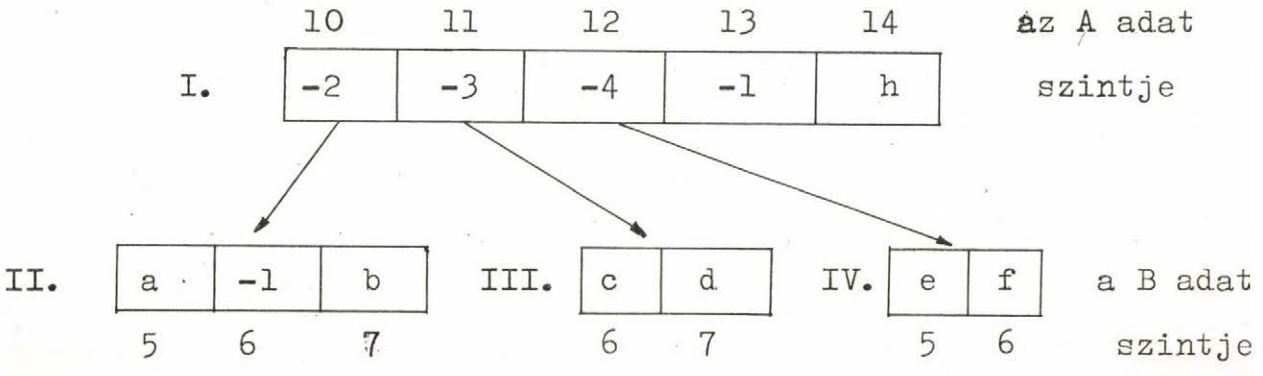
Egy megjegyzés az itt felsorolt szabályokkal kapcsolatban: A viszonylag szigorú megkötések /pl. a táblázatok sorszámozásának gráfok és szintek szerinti rendje, a gráf irányitásának monotonitása, stb./ a biztonságos felhasználást és a jobb áttekinthetőséget segitik elő.

Tekintsünk egy példát az előzőkben definiált döntési gráf kitöltésére. Egy $f(A,B)$ függvényt adunk meg a $10 \leqq A \leqq 14$, $5 \leqq B \leqq 7$ intervallumon /A és B egész érték/. A függvényt mondjuk az alábbi értéktáblázattal irjuk le /1. táblázat/.

B \ A	10	11	12	13	14
5	a	-1	e	-1	h
6	-1	c	f	-1	h
7	b	d	-1	-1	h

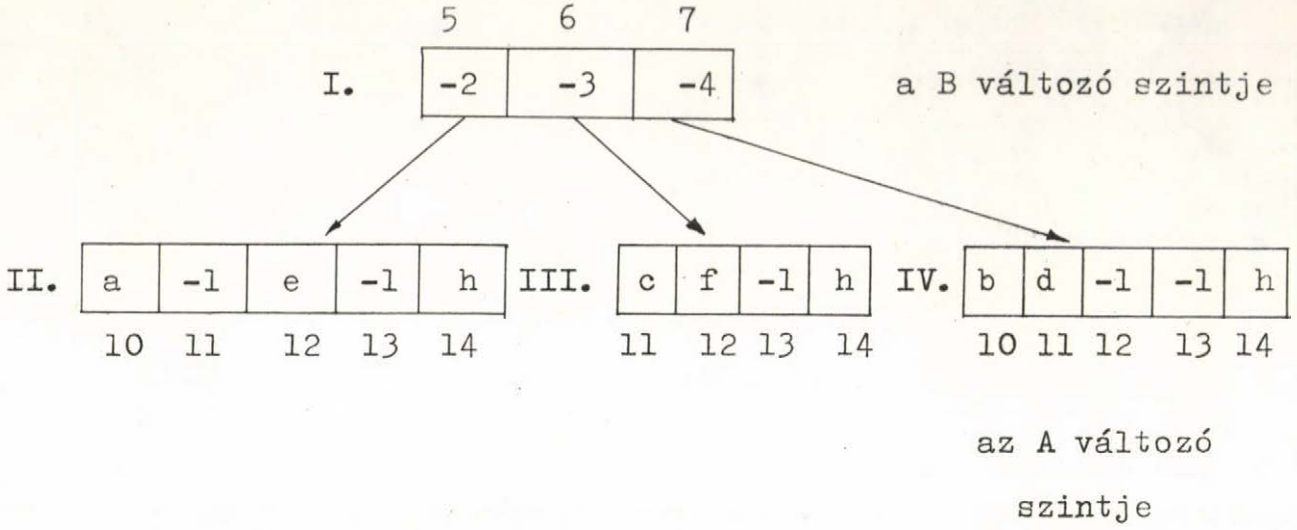
1. táblázat

A függvényértékek: a,...,h és -1. A -1 érték itt is hibát jelez. Egy az itt definiált függvénynek megfelelő döntési gráfot mutat be a 3. ábra./A táblázatsorszámok római számok, a táblázatkockák mellé irt értékek a megfelelő változóértékek - relativ cimek./



3. ábra

Látható, hogy a második szinten váltakozva két és három elemü táblázatok vannak, attól függően, hogy az A változó mely értékéhez kapcsolódnak. Megjegyzendő, hogy az A=13 és A=14 esetek előfordulási gyakorisága nagymértékben befolyásolja az átlagos vizsgálati lépésszámot /ilyenkor ugyanis csak egy lépéses az eljárás/. Ugyanezt a függvényt egy másik döntési gráffal is definiálhatjuk /felcserélve a két változó sorrendjét/ - 3/a. ábra.



3/a. ábra

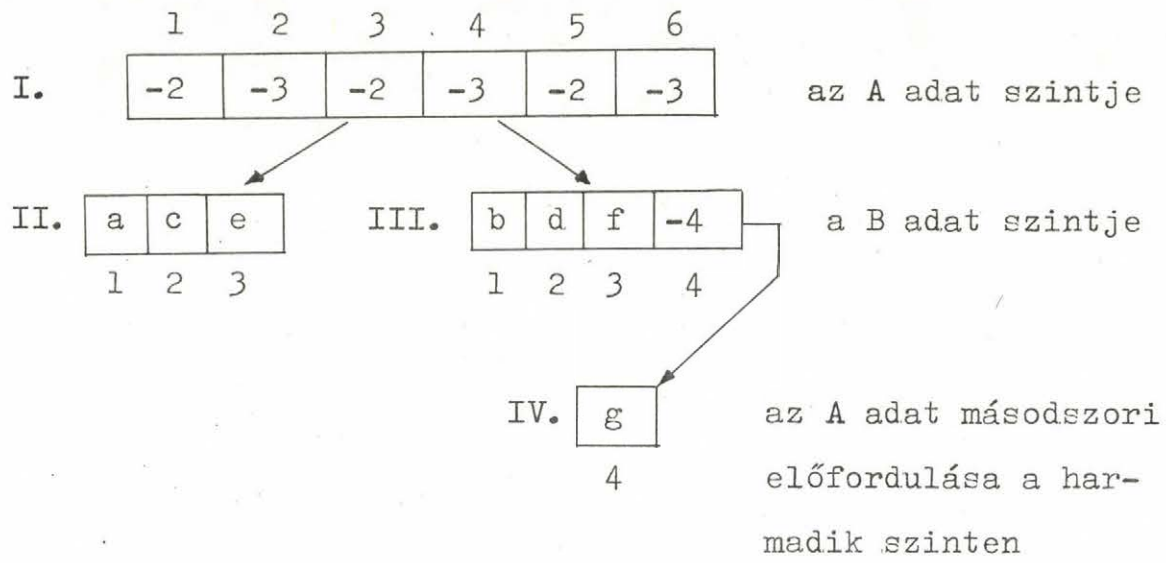
Az utóbbi megoldásmód kevésbé hatékony mint az első /3. ábra/. Az első esetben 12 rekesz, az utóbbiban 17 rekesz szükséges a döntési gráf tárolására /a gráf leírásához szükséges segédváltozókra most nem vagyunk tekintettel/. Az átlagos átfutási idő is hosszabb az utóbbi esetben /3/a.ábra/, hiszen minden vizsgálat kétlépéses /mig az első esetben voltak egylépéses utak is - A=13,14/.

Bemutatunk még egy olyan gráfot is, amelyben egy adat két különböző szinten is szerepel. Az f(A,B) függvény értéktáblázata:

B \ A	1	2	3	4	5	6
1	a	b	a	b	a	b
2	c	d	c	d	c	d
3	e	f	e	f	e	f
4	-1	-1	-1	g	-1	-1

2. táblázat

Egy megfelelő döntési gráf /4. ábra/:



4. ábra

3.1.3 A feladatok összeállitása, futtatása

Mint már az előzőkben emlitettük, a kódoló, ellenőrző eljárást egy szerkesztő /generáló/ program állitja elő minden egyes futásnál újra és újra /erről a programozási technikáról ld. pl. [10], [13]/. A szerkesztő programmal kell közölni, hogy milyen tipusú és méretü feladatokat kivánunk elvégezni. A szerkesztő eljárás az igények alapján állitja össze a feldolgozó

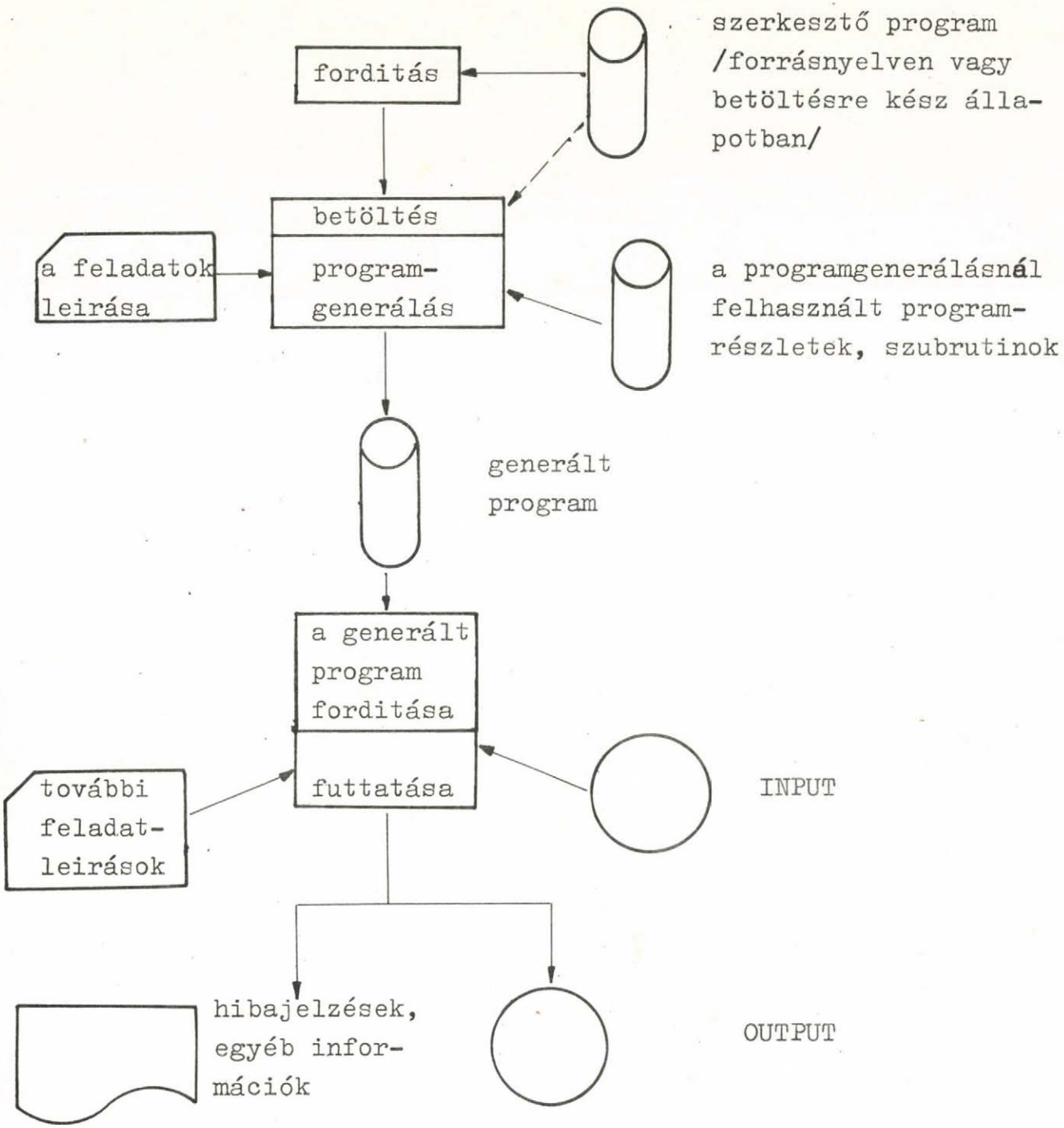
programot. A szerkesztő program számára kell megadni azokat a FORTRAN nyelvü eljárásokat is, amelyek az ellenőrzési kódolási rendszerbe - az egyszerü korlátellenőrzések és a döntési gráfok mellé - beépithetők /ld. 3.1.1 pont/.

Az igy létrejött ellenőrző kódoló programban paraméterekkel adhatjuk meg az input output rekordok formáját, a hibajelzés módját, a korlátellenőrző eljárásokat és a döntési gráfokat. A kódolási folyamat olyan, hogy egy előző lépésben létrehozott adat a következőkben felhasználható.

A döntési gráfok leirását nagymértékben segiti a 3.10 pontban leirt adatbeviteli lehetőség /ZSAK szubrutin/.

A feladat gépi futtatását az 5. ábra szemlélteti. Lényegében ugyanez a folyamat játszódik le egyébként a rendszer többi szerkesztő-szerkesztett programjának alkalmazásakor is.

Az ellenőrzési, kódolási eljárásokat több lépésben irhatjuk le. Egy lépésben csak egyféle tipusú leirást /vagy korlátellenőrzést vagy döntési gráfot, vagy a felhasználó által irt programrészlet aktivizálását/ adhatunk meg. Egy lépésben több korlátellenőrzést is igényelhetünk, döntési gráfot azonban egyszerre csak egyet definiálhatunk.

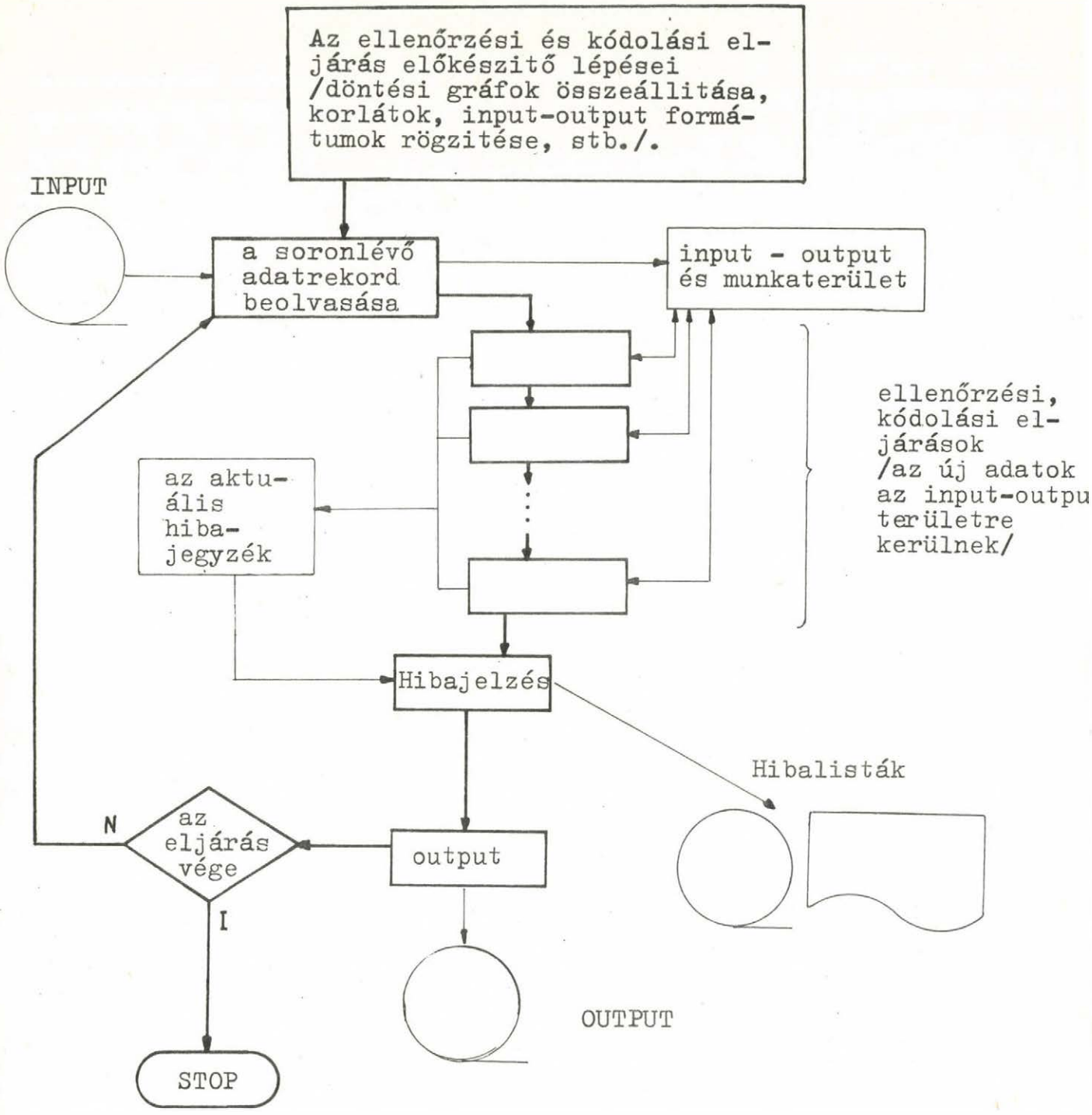

szerkesztő program
/forrásnyelven vagy
betöltésre kész állapotban/
fordítás
betöltés
a feladatok leirása
program-
generálás
a programgenerálásnál
felhasznált program-
részletek, szubrutinok
generált
program
a generált
program
fordítása
futtatása
további
feladat-
leirások
INPUT
hibajelzések,
egyéb infor-
mációk
OUTPUT

5. ábra

Közvetlenül egy döntési gráf leirása után jelezhetjük, hogy ha ugyanazt az eljárást más adatokon is el kivánjuk végezni. Egy példa erre a lehetőségre a kórházi morbiditási vizsgálatból: a kórházi adatlapon több diagnózis is szerepel - a beutaló, az ápolást indokló, többféle kisérő vagy következményes betegség, a halál oka. Ezek mindegyikénél sok azonos ellenőrzést /összeférhetetlenségi vizsgálatot/ és kódolást /az ugynevezett röviditett jegyzékekre való leképezést/ kell elvégezni. A diagnózisok közül elegendő csak az egyikre megadni a szükséges ellenőrzéseket, kódolásokat, a többinél csak hivatkozni kell az elsőként megadott döntési gráfra.

A felhasználó által leirt, és a szerkesztő program által beépitett programrészletek /ellenőrzési, kódolási feladatok/ gyakorlatilag tetszőleges FORTRAN nyelvü leirások lehetnek. Ezek a programrészletek - ugyanúgy mint a korlátellenőrzések és a döntési gráfok vizsgálata - minden input rekord feldolgozásánál újra és újra futnak.

Ezt a kapcsolatrendszert - a programlépések kapcsolatát és az adatáramlást - mutatja be a 6. ábra.


Az ellenőrzési és kódolási eljárás előkészitő lépései /döntési gráfok összeállitása, korlátok, input-output formátumok rögzitése, stb./.
INPUT
a soronlévő adatrekord beolvasása
input - output és munkaterület
ellenőrzési, kódolási eljárások /az új adatok az input-outpu területre kerülnek/
az aktuális hiba-jegyzék
Hibajelzés
Hibalisták
N
az eljárás vége
I
output
OUTPUT
STOP

6. ábra

Az ábrán szereplő előkészitő fázis egyik legfontosabb lépése az, hogy a program kiirja a feldolgozásban szereplő döntési gráfok által definiált megengedett és hibás értékkombinációkat, és az első esetben az új kódértékeket is. Az 1. táblázatban megadott f(A,B) függvény esetén például a következő listát kaphatjuk /3. táblázat/.

Változók: A,B

Hibás kombinációk		új értékek	jó kódkombinációk	
		a	10	5
10	6			
		b	10	7
11	5			
		c	11	6
		d	11	7
		e	12	5
		f	12	6
12	7			
13				
		h	14	

3. táblázat

Az A=13,14 esetben a függvényérték független a B változó értékétől, ezért annak értéke nem kerül kiirásra. A program által kiirt táblázatban az A,B,a,...,h jelek helyett természetesen konkrét sorszámok /az A,B adat sorszáma/ és számértékek szerepelnek.

3.2 Mintakiválasztás, mintaellenőrzés

Statisztikai feldolgozoknál központi szerepet játszik a vizsgált minta kiválasztása, a megfelelő mintanagyság meghatározása. A SIS77 rendszerben lehetőség van arra, hogy előre rögzitett elemszámú részmintákat válasszunk ki a következő módon:

Kijelöljük a mintaelemek **adatainak** egyikét /egy rekordelemet/. Ennek a kijelölt adatnak minden egyes lehetséges értékére megadjuk a kivánt mintaelemszámot. A kórházi morbiditási vizsgálatban ez az adat a szakma /a kórházi osztály/. Szakmánként kijelölhetjük, hogy hány beteg adatát akarjuk a részmintába bevenni. Ha ismerjük a teljes populáció - ezen adat szerint bontott - részpopuláció létszámait /az egyes kórházi szakmák összesitett adatait/, akkor olyan mintaelemszámokat

adhatunk meg, amelyek az egyes részpopulációkban /kórházi szakmákban/ adott arányú mintát biztositanak.

A SIS77 mintakiválasztó programjában a minta felbontását - az előbb leirt módon - meghatározó adat mellett egy másik paramétert is meg kell adni. Ennek az adatnak az értékei szerint döntünk, hogy mely egyedet választjuk be a mintába. A beválasztás egy rögzitett sorrend szerint történik. A kórházi morbiditási vizsgálatban a beteg születésnapja alapján történik a kiválasztás. Először minden 4-én született beteg kerül a mintába. Ha ez nem elegendő a kivánt mintaszámhoz, akkor rendre a 14-én, 24-én, stb. született betegek közül kell választani. Ez a módszer azért jó, mert igy más lényeges adatoktól független szempont alapján tudunk mintát képezni. Ha egy adatfelvételnél nem áll rendelkezésünkre ilyen adat /mint pl. a beteg születésnapja/, mely nyilvánvalóan független a vizsgálat tartalmától /pl. a megbetegedésektől, a kórházak terhelésétől, a beteg foglalkozásától, stb/, akkor mondjuk az ellenőrző, kódoló fázisban épithetünk be egy "véletlenszám generátort", amely a kiválasztás alapjául szolgáló adatot előállitja.

A kialakitott mintával kapcsolatban sokszor szükségünk van előzetes információkra, elsősorban egy-két változó szerinti gyakoriságeloszlásokra. Például a tényleges feldolgozás meginditása előtt tudni akarjuk, hogy valamennyi kórház valamennyi osztálya beküldte-e a kivánt nagyságú mintát, stb. Ezt a feladatot a SIS77-ben egy paraméterekkel vezérelhető program látja el, amely a karakter formájú input rekord tetszőleges változóiból készit gyakoriságeloszlásokat /legfeljebb 3 változóra/. Ennek a programnak az alkalmazása nagyon egyszerü, de mivel viszonylag időigényes, nem célszerü vele részletesebb statisztikákat készitení.

3.3 Szelekció, konverzió

Ebben a rendszerrészben lép a feldolgozás egy olyan fázisba, amelyben már jelentős mértékben megjelennek a SIS77 speciális tulajdonságai. Itt hivjuk fel a figyelmet arra, hogy elsősorban az ellenőrző, kódoló eljárás, de a mintakiválasztás és a mintaellenőrzést szolgáló gyakoriságszámoló program /ld. 3.2 pont/ is bármely más rendszerben felhasználható, amelyben karakter formájú fix rekordok /vagy fix részekből felépithető rekordok/ szerepelnek.

A szelektáló, konvertáló részrendszert /ugyanúgy mint az ellenőrző, kódoló programot/ egy szerkesztő programmal állithatjuk össze /ld. 5. ábra/.

Vegyük most sorra a szelektáló, konvertáló részrendszer funkcióit!

3.3.1 Adminisztráció

Ez a részrendszer és a következőkben leirtak is adminisztrálják a létrehozott file-okat, és az input file-okat is a rendszeradminisztráció alapján tudjuk feldolgozni - ld. 3.9 pont. A szelektáló konvertáló részrendszer abban különbözik a SIS77 többi részétől, hogy itt lépnek be a rendszerbe a karakter formájú alapadatok, itt lesznek először adminisztrálva /és konvertálva/. Ekkor az input file-ra vonatkozó információk még nincsenek a rendszerben, ezeket a felhasználónak kell megadnia.

3.3.2. Input lehetőségek

Ez a részrendszer többfajta inputot is feldolgozhat:

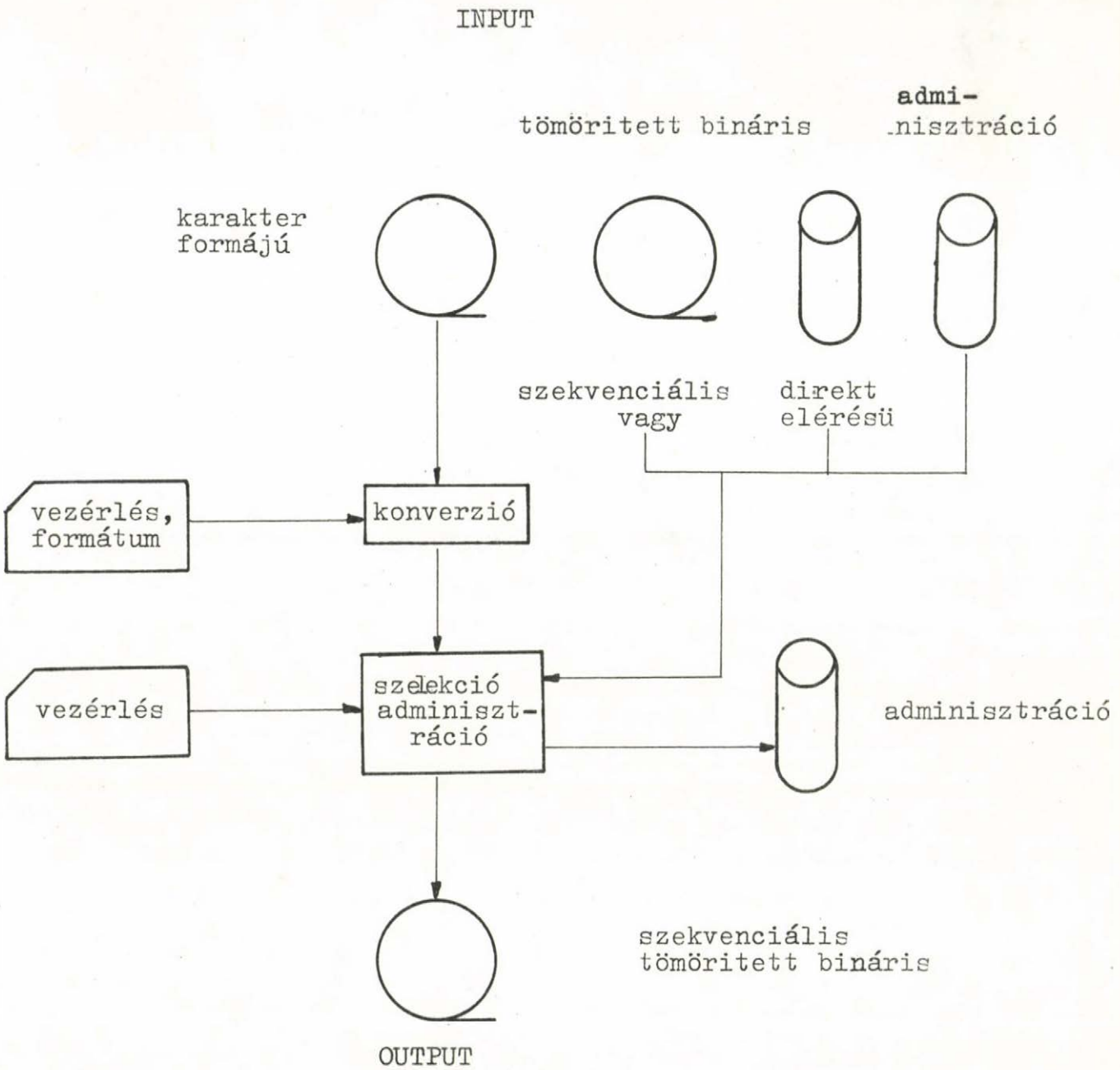
/1/ Karakter formájú input. Ilyenkor a rendszer a kijelölt szelekciós müveleteken túl egy tömöritett

bináris formába konvertálja a kiválogatott adatokat. Ennél a tárolásmódnál egy szóban több adat is elhelyezhető. Egy adat annyi bitet foglal el, ahány bit az adat maximális értékének tárolásához szükséges. Karakter formájú inputnál az input file nincs adminisztrálva. Ekkor a felhasználó irja le a szükséges file-ok /input-output/ valamennyi jellemzőjét /ld. 3.3.1 pont/.

/2/ Tömöritett bináris input, szekvenciális file-ról. Ezek a SIS77 standard, adminisztrált file-jai.

/3/ Direkt elérésü input file. Ez a file is tömöritett bináris formában tárolt adatokat tartalmaz. Egy fej rekorddal van ellátva /ld. 3.6 pont/, amely mutatókat tartalmaz. Ez a mutatótáblázat kisméretü részpopulációk gyors kiválasztását teszi lehetővé.

Az /1/-/3/ feldolgozásmódot szemlélteti a 7. ábra.


INPUT
tömöritett bináris
admi-
nisztráció
karakter
formájú
szekvenciális
vagy
direkt
elérésü
vezérlés,
formátum
konverzió
vezérlés
szelekció
adminiszt-
ráció
adminisztráció
szekvenciális
tömöritett bináris
OUTPUT

7. ábra

3.3.3 Válogatási lehetőségek

A végrehajtható szelekciós müveletek a következők:

/1/ Kiválaszthatjuk az input rekord számunkra szükséges adatelemeit, és az output rekordon már csak ezek az elemek - megadásuk sorrendjében - jelennek meg.

/2/ Az egyes rekordokra - FORTRAN nyelven - logikai szabályokat irhatunk elő, és a szabály teljesülése esetén kihagyjuk /vagy megtartjuk/ a soronlévő rekordot. A logikai szabály mindig a soronlévő rekord adatértékeire vonatkozhat. A felhasználó által leirt logikai kifejezés a szerkesztő program által automatikusan beépül a feldolgozási programba.

/3/ Bonyolultabb - nagyon sok különböző értékre vonatkozó - logikai kifejezés leirása a hagyományos eszközökkel nehézkes. Ilyen esetekre dolgoztunk ki egy jól használható módszert - ld. [4] - [7]. Az előző pontban már emlitett adatbeviteli eljárás /ld. 3.10 pont/ segitségével a felhasználó táblázatokban tüntetheti fel, hogy mely adatértékek megengedhetők és melyek nem. Az egyes elemi táblázatokat - amelyek

egy feldolgozási lépésben a pillanatnyi adatértékek függvényében "igaz" vagy "hamis" logikai értékeket képviselnek - a felhasználó tetszőleges /konjunktiv vagy diszjunktiv normálformára hozott/ logikai kifejezésbe foglalhatja. Az adatértékek eloszlása /együttes eloszlása/ ismeretében lehetőség van a feldolgozás futásidőben való optimalizálására is. Ezeket a kérdéseket bővebben tárgyalják a fent emlitett [4] -[7] publikációk.

/4/ Kisebb részpopulációk - például egy kórház beteganyagának - kiválasztása esetén gazdaságtalan a teljes populációt végigvizsgálni. Ezért tettük lehetővé azt, hogy néhány /egyszerre legfeljebb négy/ adat értéke /értékkombinációja/ szerint közvetlenül is elérhessük az adott értékhez /értékkombinációhoz/ tartozó rekordokat. Erre a célra egy olyan direkt elérésü file-t kell képezni, melynek elején az egyes részpopulációk helyét meghatározó mutatók vannak /ld. 3.6 pont/. A felhasználó a mutatótáblázat értékeire vonatkozó /FORTRAN nyelven irt/ logikai kifejezés megadásával kijelöli, hogy milyen adatértékhez tartozó részpopulációra van szüksége, majd a szokásos módon /mint a fenti /1/ - /3/ esetben/ leirja a válogatási eljárást. A szelekció ekkor két

részre bomlik. Első lépésben a mutatótáblázat alapján a program megkeresi a kijelölt részpopulációt tartalmazó file-terület kezdőpontját /kezdőpontjait/. Ezután a kijelölt területen lévő rekordokat vizsgálja a szelekcióra megadott logikai kifejezés szerint.

3.3.4 Output

A válogató, konvertáló eljárás outputja mindig egy tömöritett bináris formájú szekvenciális file. Az input file rendezettségét az output file megtartja, és ez ki is használható - ha az output rekordba átkerülnek a rendezési kulcsok.

3.3.5 A program alkalmazásának célja

A szelektáló, konvertáló program bizonyos értelemben a SIS77 központi része /jól mutatja ezt az 1. ábra és a [13] tanulmány rendszerábrái is/. Ezen a ponton bonthatjuk részekre a feldolgozandó adatrendszert. Tipikus jelenség az, hogy a statisztikai feldolgozásokon belül elkülönithetők olyan információszolgáltatási egységek, amelyek az alapadatoknak csak bizonyos részét használják fel. Célszerütlen tehát minden egyes feldolgozási lépésben a teljes adatrendszert vizsgálni. Egy vizsgálati

csoporthoz szükséges adatrekordokat illetve rekordelemeket különválaszthatjuk, igy a felhasznált tárolóterületet és a feldolgozási időt is csökkentjük. Ha például csak a budapesti betegeket akarjuk vizsgálni, akkor adatáikból egy külön file-t hozunk létre és a továbbiakban ezzel, az eredetinél lényegesen kisebb méretü file-lal dolgozunk. Az egyedek szétválasztására - a kisebb file-méretekből eredő előny mellett - még azért is szükség van, mert jelentős kapacitást köt le az is, ha a felesleges egyedeket a különböző feldolgozások folyamán újra és újra el kell különiteni.

3.4. Rekordok összevonása

Statisztikai adatszolgáltatást nem mindig az alapsokaság egyedein kell végezni. Esetenként nem egy-egy mintaelem a feldolgozás egysége, hanem több mintaelemet össze kell vonni, vagy egy elemet több részre bontani. Most az előbbi feladattal foglalkozunk.

A kórházi morbiditási adatfeldolgozás alapját az ápolási esetek /egy beteg egyszeri kórházi ápolása/ alkotják. Egyes statisztikákban azonban nem az esetek számát, hanem az ápolásban részesült személyek számát kell feltüntetni. Ilyenkor az egyes személyek

különböző ápolási eseteit egyetlen rekordba vonjuk össze. Általánosan fogalmazva, a feladat az, hogy a statisztikai sokaságot alkotó rekordok egyes elemeit azonositónak tekintve, az egybetartozó rekordokat egyetlen rekorddá egyesitjük. A SIS77-ben az egyesitett rekordok - az őket alkotó rekordok számától és tartalmától függetlenül - fix szerkezetü rekordok, azért, hogy a statisztikai feldolgozás egységes menetébe beilleszkedhessenek.

Az egyesités a következőképpen történik: Kijelöljük a rekordazonositó elemeket /pl. a kórházban ápolt beteg személyi adatait/. Ezek szerint rendezzük a file-t. A rendezett file-on egymás mellé kerülő, összetartozó rekordokból egy rekordot képez a rendszer a következő módon: Kijelöljük, hogy a rekord mely elemei /és milyen sorrendben/ kerülnek át az egyesitett rekordba. Az összetartozó rekordok közül mindig az elsőnek az adatértékei kerülnek át az új rekordba. A kijelölt adatokon túl az összetartozó rekordok száma is - mint új adat - megjelenik az output rekordban. Emellett lehetőség van egy további adat kumulálására is /például gyüjthetjük egy személynek egy éven belüli összes ápolási idejét/.

A feldolgozó programot ennél a feladatnál is egy szerkesztő program állitja elő.

Az input file tömöritett bináris formájú rendezett file. A rendezést a SIS77-ben a felhasznált gép /HWB 66/60/ SORT programjával oldhatjuk meg.

Az output file ugyancsak tömöritett bináris file, és megtartja a rendezést, ha a szükséges azonositó elemeket /melyek most rendezési kulcsok/ átvisszük az output rekordba.

Mind az output, mind az input file adminisztrált file /ld. 1. ábra/.

Ennek a rendszerprogramnak /rekordok összevonása/, és a következő 3.5 pontban leirtnak a célja elsősorban az, hogy egységes adatfile — formát adjon a statisztikai feldolgozások számára. Ez a standard forma mindig tömöritett bináris forma és mindig a feldolgozás egységeit /ápolási eset, személy, stb./ tartalmazza rekordként. A rekordok mindig fix formátumuak.

Ez és a következő pontban /3.5/ szereplő eljárás - bár bizonyos szempontból speciális fileokat hoz létre - általános adatbázisszervezési feladatot old meg.

3.5 Rekordok felbontása

Ez a program is ugyanolyan feltételek mellett

müködik mint az előbb leirt /rekordösszevonó/ program /tömöritett bináris input-output, file-adminisztráció, a programot szerkesztő program állitja elő, az új rekordokba csak a kijelölt elemek kerülnek át/. Két eltérés van az előbbi és a mostani feladat között. Egyrészt a rekordfelbontáshoz nincs szükség az input file rendezésére, másrészt nincs értelme a rekordismétlési számnak és kumulált értéknek. Ehelyett jelezhető az, hogy hányfelé lett bontva az eredeti rekord.

Az elvégzendő feladat a következő: Ha egy rekordban több olyan adat van /vagy lehet/, amelyek nem megkülönböztethetők, akkor az ilyen rekordokat a rendszer több példányban állitja elő, úgy hogy az új rekordban már csak egy-egy szerepel a meg nem különböztethető rekordelemek közül. /Megjegyezzük, hogy az utóbbi két eljárás - rekordegyesités illetve felbontás - szoros kapcsolatban áll a Codd-féle relációs adatmodellel, ld. [13]./

Vegyünk egy példát a kórházi morbiditási vizsgálatból! Egy kórházi ápoláskor többféle diagnózis is rákerül a beteg adatlapjára. Ezek a diagnózisok részben teljesen egyenrangúak /kisérő illetve következményes betegségek /, részben a számuk sem rögzitett /legfeljebb az adatlap terjedelme ad egy felső

korlátot/. A most tárgyalt rendszerprogram segitségével az ápolási esetek rekordjait annyi példányban megismételhetjük, ahány kisérő /következményes/ betegség van feltüntetve az adatlapon. Igy az uj rekordok mindegyike egy kisérőbetegséget tartalmaz. Ha nincs kisérőbetegség vagy ha csak egy van, akkor megmarad az eredeti rekord. Igy egy egységes formában kezelhető rekordsorozatot kapunk. Ezzel a kérdéssel - a relációs adatbázis-modellek normálformáival kapcsolatban - a [13] tanulmány foglalkozik részletesebben.

3.6 Direkt elérésü file-ok

Direkt elérésü file-oknak a SIS77-ben történő alkalmazását a 3.3 pontban irtuk le. Itt csak az előállitás módjáról beszélünk.

Ezek a file-ok standard SIS77 file-ok /tömöritett bináris forma, adminisztrált file/, azzal az eltéréssel, hogy a file elején egy kulcstáblázat van. A file előállitását végző programot ebben az esetben is egy szerkesztő program készíti el.

A felhasználó a feladat leirásakor megadja, hogy hány szempont szerint kivánja az output file-on közvetlenül elérni az egyes részpopulációkat. Azt, hogy melyek ezek a szempontok /amelyek szerint a populáció részekre bontható,

illetve részenként közvetlenül elérhető/ nem kell megadni, mivel ezek az adatok az input file rendezési kulcsai. A rendezési kulcsok viszont már az input file adminisztrációjában leírásra kerültek.

A rendezettségre egyébként egyrészt azért van szükség, hogy ne egy láncolással kelljen az összetartozó adatokon végighaladni, másrészt nagytömegü adat olvasásánál lényeges szempont az, hogy az egyidejüleg beolvasott adatok lehetőleg fizikailag is egy helyen legyenek.

A program a vezérlő adatok és az input file feldolgozása után összeállit egy mutatótáblázatot. A táblázat dimenziószáma a felhasznált rendezési kulcsok számával egyezik meg, és legfeljebb 4 lehet. A dimenziók sorrendje megegyezik a kulcsok sorrendjével. A táblázatban egy kulcskombinációnak megfelelő indexü helyen az a cim /rekordszám/ van, amelyen az adott kulcskombinációt tartalmazó rekordok közül az első szerepel. Igy egy részpopuláció kiválasztásához a viszonylag kisméretü mutatótáblázatban kell kikeresni a megfelelő tárolócimet /cimeket/.

Példaként vegyük azt az esetet, amikor néhány kórház néhány kijelölt osztályát akarjuk vizsgálni. A kórház és osztálykód szerint ren-

dezzük a felhasználandó adatállományt. A rendezés után a felhasználó kell gondoskodjék a rendezett file adminisztrációjáról /ld. 3.9 pont/, különben a direkt elérésü file-t előállitó program nem fogadja el inputként. Az output file elején lévő mutatótáblázat dimenziói a jelen példában a kórházkód és az osztálykód. A mutatótáblázat beolvasásáról, az adminisztrációban rögzitett paraméterek alapján a feldolgozó program automatikusan gondoskodik.

3.7 Táblafile-ok készitése

Mig a szelektáló és konvertáló részrendszer egy gazdaságos tárolási forma létrehozása mellett az adatrendszer részekre való bontását oldotta meg, addig a "táblafile" készités hasonló célokat szolgál, de egészen más módon.

A leglényegesebb különbség az, hogy a tárolásmód megváltoztatása nem csak formai /kódolás/ hanem minőségi változást is eredményez. Nem egyedek adatait hanem tipusokra vonatkozó értékeket tartalmaz ez a file. A standard, tömöritett, bináris file-okból - melyek egyedekre /pl. ápolási esetekre/ vonatkozó

adatokat tartalmaznak - statisztikai táblázatokat hozunk létre /ld. 1.1 és 2.2 pont/. Ezek a táblázatok a statisztikai vizsgálat tárgyát képező típusok adatait tartalmazzák. A táblázatokat az első lépésben nem nyomtatjuk ki, hanem permanens file-ra helyezzük - ezt a file-t nevezzük táblafile-nak /transzformált file-nak, tipusok filejának, statisztikai file-nak/.

A táblázatformából adódó szabályosságok a táblafile gyors feldolgozását teszik lehetővé. A táblafile azonban, a standard bináris file-okkal ellentétben, nem dolgozható fel újra és újra, csak a táblázatkiiró program inputjaként szerepelhet /ld. 1. ábra/. A táblafile készitése egyben egy adatválogatást is jelent. Egy táblafile-ban, a jelenlegi rendszerben, legfeljebb nyolc bontási szempont /dimenzió/ szerepelhet, mig egy standard, tömöritett, bináris file húsz különböző adatot is tartalmazhat /ez utóbbiak közül kell kijelölni a táblafile dimenzióit/.

A táblafile-ok alkalmazásának előnyeit a [13] tanulmány részletesen tárgyalja.

A táblafile-t előállitó program müködésével kapcsolatban a következő emelhető ki.

A feldolgozó programot szerkesztő program állitja össze. A szerkesztésnél határozható meg az elvégzendő feladat mérete és tipusa.

Háromféle módon állithatunk elő táblafile-t. Mindhárom esetben a központi memóriában készül a táblázat, de mig az első esetben az egész táblázat az operativ tárban van, addig a másik két esetben vagy egy rendezett input file-t dolgozunk fel, és igy lehetőség van a táblafile részletekben történő elkészitésére /vagyis a teljes táblafile méretnek csak egy törtrészét foglaljuk le a központi tárban/, vagy az input file-t ismételten többször is beolvassuk és igy bontjuk részekre a táblafile elkészitését. A két utóbbi esetben az az adat amely szerint felbontjuk a teljes táblázatot /rendezett file esetén ez a rendezési kulcs/, a táblafile "legkülső" dimenziója lesz.

A táblafile készitésénél összevonásokat is alkalmazhatunk. Ezeket az összevonásokat leiró kódtáblákat a rendszer más részeiben is használt adatbeviteli eljárás segitségével tölthetjük ki /ld. a 3.10 pontban a ZSAK szubrutin leirását/.

A táblafile elemei gyakoriságokat, és az input file egy kijelölt adatának összegzett értékeit tartalmazzák /egy gépi szóban - ld. 2.2 pont/.

A feldolgozó program a táblafile, illetve az összevonásokat leiró táblázatok méretétől függően változó nagyságú memóriát igényel. Ugyancsak változó méretü lehet az output file blokkmérete. A változó méretü adattömböket /ugyanúgy mint pl. az ellenőrzo, kódoló eljárásnái/ a szerkesztő program deklarálja.

A szerkesztő program alkalmazása a táblafile készitésénél különösen jelentős, mert a feldolgozásnak ez az a pontja, ahol még magytömegü alapadattal /a vizsgált egyedek adataival/ kell dolgozni, tehát egy viszonylag számolásigényes müveletet /adatok beolvasása, indexszámitás, gyakoriságok és összegek gyüjtése, rendezési feltételek vizsgálata/ kell nagy tömegben elvégezni. Ugyanakkor erre a feldolgozási lépésre viszonylag gyakran kerül sor. A szerkesztő program elsősorban a táblaindex számitását optimalizálja /ld. [10], [11]/, de a tömöritett bináris file-ok feldolgozását is hatékonnyá teszi.

A táblafile - és a készitésénél felhasznált input file is - adminisztrált file.

3.8 Táblázatok kiirása

A táblázatkészitő program az 1.2 pontban /illetve a [13] tanulmányban/ már vázolt feladatokat látja el. A táblázatok kiirásakor kisérőszövegeket /fejlécek, adatnevek/ ir ki, járulékos adatokat /százalékok, átlagok, részösszegek, stb./ számit és grafikont készit.

A programnak két változata van. Az egyik kifejezetten terminálon /képernyőn/ való táblakiirás célját szolgálja.

3.8.1 Táblakérdezés terminálon

Ez a feladat az eredeti táblázó program egy leegyszerüsitett változatával oldható meg. Elsősorban kisméretü táblázatokat célszerü ilyen módon előállitani /a képernyőn való megjelenités terjedelmes output tanulmányozására nem alkalmas/. A képernyőn adott méretkorlátok miatt a táblázat csak a következő értékeket tartalmazza: gyakoriság, egy kódösszeg, százalékok, kódösszeg átlag. Grafikont nem lehet készitени. A programot nem szerkesztő program állitja elő, igy a táblaméretek maximuma rögzitett. A felhasznált input file-t - amely mindig egy táblafile - nem szükséges adminisztrálni.

A program nemcsak terminálról, hanem kötegelt feldolgozásmódban is futtatható.

3.8.2 A táblázatkiirás sornyomtatón

A szokásos sornyomtató méretek /lapszélesség, magasság/, és az eredmények nyomtatásban történő rögzitése összetettebb és terjedelmesebb output formát és tartalmat tesz lehetővé. A táblázatkiiró program alapváltozata ezért egy szerkesztett program formájában müködik, hogy a terjedelmes memóriatömbök /a kinyomtatásra kerülő táblázat, a táblázatban szereplő névjegyzékek és az esetleges összevonásokat definiáló táblák/ dinamikus deklarációja lehetővé váljék. Igy a kiirandó tábla méretét csak a rendelkezésre álló memória korlátozza, illetve kisebb táblázatoknál csak a szükséges méretü memória lesz felhasználva.

Ez a programváltozat csak adminisztrált táblafile-t fogad el inputként. A táblázó eljárás felhasznál még segédfile-okat is /ld. 1. ábra/, de ezek nem adminisztrált file-ok.

A segédfile-okon vannak a táblázáshoz szükséges szövegek /fejlécek, adatnevek/ és a táblázatba

kerülő statisztikai viszonyszámok /pl. a kórházi morbiditási vizsgálatban a 10 ezer lakosra jutó eset és nap/ számitásához szükséges statisztikai alaptáblázatok /pl. népességstatisztikai adatok/. A viszonyitásban felhasznált táblázatok lehetnek például egy előző feldolgozás eredményeként nyert gyakoriságtáblázatok is /pl. két különböző időszak adatait akarjuk összehasonlitani/. Fontos megjegyezni, hogy az összehasonlitásban szereplő táblázatoknak dimenziószámban és méretben nem kell megegyezniük. Például egy életkor szerint bontott táblázat kiirásakor felhasználhatunk életkor, foglalkozás és iskolai végzettség szerint bontott viszonyszámokat. Az utóbbi két bontási szempont felesleges. A táblázó program a felhasználó által adott vezérlő paraméterek alapján automatikusan elvégzi e két bontási szempont összevonását.

Ugyanilyen összevonásokra kerül sor a táblafile-nak a nyomtatni kivánt formára történő leképezésekor is.

3.8.3 A táblázatleképezési eljárás

A táblafile-ok alkalmazásának jelentősége a már leirt szempontokon kivül /3.7 pont/, abban

rejlik, hogy viszonylag nagyméretü táblafile is rendkivül gyorsan leképezhető a felhasználó által kivánt táblázatra. Ennek a feladatnak a megoldására a szokásos programozási nyelvek egy hatékony és egyszerü eszközt biztositanak. Ezzel a módszerrel kapcsolatos korábbi eredményekre hivatkoztunk a bevezetésben, a [3] dolgozat emlitésekor. A módszer a következő:

A táblafile egy n-dimenziós táblázat. Dimenzióit d_i-vel jelölve, legyen $a_i \leqq d_i \leqq b_i$, i=1,...,n. Az a_i, b_i korlátok és a d_i dimenziók egész értékek. Ebből az n-dimenziós táblázatból akarunk egy m-dimenziós táblázatot készitени /$m \leqq n$/ úgy, hogy közben az egyes d_i értékeket D_i értékekre képezzük le, $A_i \leqq D_i \leqq B_i$ /szintén egész értékek/.

A leképezés két részre bontható. Az első részben egyes dimenziókat kihagyunk /ha $m < n$/. Itt alkalmazhatjuk a bevezetésben emlitett módszert. A leképezést végző programban n egymásba skatulyázott ciklust helyezünk. Ezek a ciklusok az input táblázat /a táblafile/ dimenzióit képviselik. A ciklushatárok az i-edik ciklusban a_i és b_i. Az input file elemeit egyenként olvassuk be a legbelső ciklusban. Az output táblázat indexét /itt mindig a memóriacimre, vagyis egydimenziós táblaindexre kell

gondolni/ a megfelelő ciklusszinteken növeljük, ha az adott sorszámú dimenziót meg kívánjuk tartani az output táblázatban. Ha az illető dimenziót el akarjuk hagyni, akkor a megfelelő szinten nem növeljük az indexet. Az egyes ciklusokhoz tartozó növekmények értéke a dimenziók sorrendjétől /a táblafile és az output tábla dimenzióinak kölcsönös kapcsolatától/ és az egyes dimenziók méretétől /b_i-a_i+1/ függ. A ciklusokon való átfutás egy adott fázisában mindig az az output index áll rendelkezésünkre, amely az éppen soronlévő input érték halmozásának helyét jelöli ki.

Ha az egyes dimenziókon belül a fent emlitett $D_i=f(d_i)$ leképezéseket is végre akarjuk hajtani, akkor a növekmények egy-egy ciklusszinten a ciklusváltozó /vagyis a soronlevő input dimenzió/pillanatnyi értékétől is függnek. Ezeket a növekménytáblázatokat a SIS77-ben a felhasználó könnyen leirhatja /ld. 3.10 pont/. Magukat a növekményeket nem is kell leirni /kiszámitani/ csak az egyes megfelelő dimenziók értékeinek leképezését meghatározó értéktáblázatot. A rendszer /a ZSAK szubrutin - ld. 3.10 pont/ a növekményeket automatikusan kiszámitja.

Ha a kiirt táblázat egyes sorai közé részösszeg-

kiirásokat akarunk illeszteni akkor ezeknek a részösszegsoroknak megfelelő sorszámú helyen üres tárolórekeszeket kell biztositani. Ezt úgy érhetjük el, hogy abban a tábladimenzióban, ahol a részösszegek megjelennek az $a_i \leqq d_i \leqq b_i$ értékeket egy nem összefüggő intervallumra képezzük le, kihagyva a létrehozott D_i értékek között a részösszegeknek megfelelő sorszámokat.

3.9 A rendszer-adminisztráció, segédfile-ok

A rendszeradminisztráció célja a különböző rendszerprogramokat összekapcsoló input-output file-ok automatikus nyilvántartása. Ezzel lehetővé válik az, hogy a felhasználó egyedül az adminisztrációs file azonositójának ismeretében pontos képet alkothasson a feldolgozás pillanatnyi állapotáról, és a feldolgozási lépések megfelelő módon kapcsolódjanak egymáshoz /egy feldolgozási lépés az előző lépésekben létrehozott file-ok közül a megfelelőt használja fel/.

A 2.2 pontban leirt file-ok /ld. 1. ábra/ közül a következőket adminisztrálja a rendszer:

Tipus	tartalom, forma
3.	Válogatott, konvertált tömöritett bináris file-ok.
5.	Direkt elérésü file-ok
6.	Speciális file-ok /rekordok összevonása, felbontása/
7.	Táblafile-ok

4. táblázat

A rendezett /szekvenciális/ file-okat /4-es tipus/ a tömöritett bináris file-okból /3-as, 4-es, 6-os tipus/ a SIS77- en kivül müködő SORT program készíti, igy a rendezés után ezeket a file-okat a felhasználó adminisztrálja.

Nem adminisztrált file-ok:

Tipus	tartalom, forma
1.	Alapfile /fix rekordok, karakter forma/
2.	Ellenőrzött minta
4.	Rendezés utáni tömöritett bináris file-ok
8.	Segédfile-ok
10.	Adminisztráló file
11.	Táblázatok
-	Programfile-ok

5. táblázat

Az adminisztráló rendszer feladata még a segéd-file-ok /ld. 2.2/ felvitele is.

Az adminisztráció lényegében két fő részre bontható:

/1/ Az adminisztráció és a felhasználó kommunikációja: új file-ok adminisztrálása /pl. a rendezés után/, felesleges file-ok törlése az adminisztrációból, a feldolgozás következő lépésben szükséges file-ok kiválasztása /pl. lekérdezhető, hogy van-e olyan táblafile, amely kórházi szakma és diagnózis főcsoport szerinti bontást tartalmaz/.

/2/ A másik rész a rendszerben létrehozott file-ok automatikus adminisztrálásából, illetve az input file-ok adminisztrációjának vizsgálatából és az input file adminisztrált paramétereinek a feldolgozási folyamatba való beépitéséből áll. Ezt a feladatot maguk az egyes rendszerprogramok végzik el.

A file-adminisztrációban rögzitett adatok: a file neve, tipusa /ld. 4. táblázat/, a file rekordelemeinek száma illetve táblafile-nál a dimenziószám, az adatelemek illetve dimenziók neve, az adatok illetve a dimenziók értékhatárai, rendezési kulcsok /ha van ilyen/, a file-on

lévő rekordok száma illetve a táblafile mérete, végül egy felhasználói megjegyzés /egy maximum 5 kártya terjedelmü tetszőleges szöveg/. Ezeknek a paramétereknek az alapján a soronlévő rendszerrész megvizsgálja, hogy a felhasználó által kijelölt input file alkalmas-e a kitüzött feladat megoldására, és ha igen, akkor beállitja az input file-nak és az elvégzendő feladatnak megfelelő paramétereket /kezdőértékeket, szerkesztett program esetén a megfelelő programutasitásokat/.

3.10 Szubrutinok

A rendszer tartalmaz néhány olyan szubrutint, amelyek több részrendszerben is felhasználásra kerülnek. Ezek elsősorban a file-adminisztrációs feladatokat látják el /ilyen feladatok az adminisztrált file-okkal foglalkozó részrendszerek mindegyikében előfordulnak/, és a megfelelő részrendszerekben ismételten előfordulnak. Ezekhez a szubrutinokhoz a felhasználó közvetlenül nem férhet hozzá.

Valamennyi részrendszerben szerepel egy szubrutin /ZSAK szubrutin/, amely a felhasználók számára is hozzáférhető, és igy nemcsak a SIS77-ben,

de más rendszerekben is alkalmazható. Ezt a szubrutint egy külön file-on tárolja a rendszer és minden alkalommal hozzácsatoljuk a soronlévő részrendszerhez /ez általában automatikusan megtörténik a szerkesztő programok segitségével/.

Ennek a szubrutinnak a feladata függvényértéktáblázatok gyors, biztonságos és könnyen kezelhető formában történő kitöltése. A felhasználó két lehetőség közül választhat: vagy maga irja le az értéktáblázatot, vagy bizonyos speciális esetekben egyes müveleteket a szubrutinra hagyhat. Az utóbbi eset többdimenziós táblázatok készitésekor, illetve egymásra való leképezésekor fordulhat elő. Többváltozós táblázatok elemeinek memóriacimét ugyanis az egyes dimenzióértékek ismerete alapján számithatjuk ki /ha nincs módunk a felhasznált programozási nyelvben többdimenziós indexezésre - pl. FORTRAN nyelven legfeljebb három indexet használhatunk/. Ezt a számitást /illetve a számitás előkészitését/ automatikusan elvégzi a ZSAK szubrutin /ld. [16] / .

A szubrutin felhasználási területei a SIS77-ben:

/1/ Az ellenőrző, kódoló rendszerben a döntési gráfok táblázatainak kitöltése. Ekkor a felhasználó

közvetlenül a függvényértékeket - hibajelzés /-1/, mutatók /-2, -3, .../ és függvényértékek /0,1,.../ - adja meg.

/2/ A szelektáló rendszerben leirható logikai kifejezések tagjait alkotó táblázatok kitöltése /"kiválasztandó", "elhagyandó" jelzések beirása/.

/3/ Dimenzióösszevonások a transzformált file /táblafile/ készitésekor. A felhasználó a dimenzióértékek megfeleltetését adja meg /az új, összevont értékek milyen eredeti értékeknek felelnek meg - pl. életkorcsoport képzés/.

/4/ Táblafile-ok leképezése a kiirandó táblázatra. A felhasználó csak az egyes dimenzióértékek megfeleltetését adja meg /pl. mely dimenziót hagyja el, melyet szerepelteti egy összevont formában és melyiknél maradnak meg az eredeti értékek/. A leképezéshez szükséges indexnövekményeket - amelyeket a táblázó program használ fel - a szubrutin automatikusan kiszámitja.

Ez a szubrutin - ugyanúgy mint a többi rendszerrész - részletesen elemzi a vezérlő kártyákat

/hibavizsgálat, hibajelzés, vezérlő kártyák listázása magyarázó szöveggel/, és a tartalmi ellenőrzés érdekében kiirathatók az elkészitett értéktáblázatok /indexnövekmény táblázatok/ is.

Ismételten felhivjuk a figyelmet arra, hogy a szubrutin a SIS77 minden kötöttségétől mentes, igy bármely kódleképezési eljárásnál - a SIS77-től függetlenül is - alkalmazható /ld. [16]/.

4. A rendszer alkalmazásának néhány technikai kérdése, javaslatok

A SIS77 jellegzetessége az, hogy a felhasználó az egyes rendszerrészeket egy önállóan kialakitott stratégia szerint alkalmazhatja, és a különböző rendszermodulokat szabadon kombinálhatja. Ezért fontos egy-egy feladat megoldása előtt a feldolgozás menetének helyes megválasztása. A jó stratégia szoros kapcsolatban van a feladat méreteivel, bonyolultságával. Esetenként más és más kombinációban kell a rendszert alkalmazni. Elképzelhető például, hogy a szelektáló konvertáló részrendszert csak konverzióra használjuk, és egyetlen tömöritett bináris file-lal dolgozunk. Ha a statisztikai sokaság speciális részmintáival dolgozunk, akkor célszerü lehet valamennyi tömöritett bináris file-t direkt elérésü formára hozni. Egyszerübb, kisméretü adatrendszernél, ha csak gyakoriságértékekre van szükségünk, az ellenőrzés és esetleges kódolás után, a 3.2 pontban leirt egyszerü gyakoriságszámoló programmal dolgozhatunk. Van azonban néhány olyan általános szempont, amelyeket a megoldandó feladat tartalmától

függetlenül érdemes figyelembe venni. Az alábbi felsorolásban két témacsoporttal foglalkozunk /4.1, 4.2 pont/. A két témacsoporton belül különböző jellegü és jelentőségü kérdéseket sorolunk fel.

4.1 Ellenőrzés, kódolás

4.1.1 Előkészitő lépések

Bár az adatfelvétel tervezésének kérdése nem tartozik szorosan a SIS77 rendszerhez, mégis röviden foglalkozunk vele. Az adatfelvétel előkészitésekor, nevezetesen a felhasznált kódrendszer kialakitásakor át kell gondolni a feldolgozás teljes menetét. Sorra kell venni a rendszerben adott lehetőségeket és korlátokat.

Nem szükséges például a kódokat rész-kódokra bontani, ha más feldolgozások ezt nem teszik szükségessé. Például a kórházi morbiditási vizsgálatban négyjegyü kórházkódot használunk /külön két jegy jelzi a kórház megyéjének kódját és külön két jegy a megyén belüli sorszámot/, holott három jegy - egy egyszerü sorszám - is elegendő a kórházak megjelölésére. Az adatszolgáltatónak a megye illetve néhány speciális intézménytipus /klinikák, országos intézetek/

kódja szerinti szétválasztás semmit se jelent, a feldolgozás során viszont automatikusan kialakithatók a sorszámokból is a bontott kódok. Elektronikus számológépen történő adatfeldolgozásnál általában nincs szükség a kódok előzetes csoportositására - ez a kézi feldolgozások illetve a Hollerith technika örökségének tekinthető. A kódok bontására ott van szükség, ahol a megjelölni kivánt objektum kódja ismeretlen, vagy nehezen hozzáférhető. Ilyenkor a kódlap lépésenként, "rákérdezéses" módon tölthető ki. Például a foglalkozáskód /ld. "FEOR"/ kitöltése történhet ilyen módon, a kódlapon szereplő különböző szintü kérdések igen-nem formában történő megválaszolásával.

Vannak azonban kikerülhetetlen problémák. Jó példa erre a betegségek nemzetközi kódrendszere. Ebben a négyjegyü kódban nem lehetséges ugyanis azt jelezni, hogy valamely esetben semmiféle diagnózis sem szerepel /pl. egy kórházi betegnél a halál oka, a beutalás indoka/. Az erre a célra legjobban megfelelő 0000 kód a kolera jelölésére van lekötve.

4.1.2 A kódolás szervezése

A rendszer első, ellenőrző, kódoló részében hozzuk létre a későbbiekben felhasznált adatokat.

Itt a felhasználónak lehetősége van arra, hogy egyrészt egy adatot többféle változatban is létrehozzon /pl. a diagnóziskódok különböző "röviditett jegyzékeit" képezze/, másrészt az adatokat az output rekordban a legmegfelelőbb sorrendben helyezze el.

A SIS77 jelenlegi változatában a szelektáló, konvertáló részben a konverzió gyorsabb, ha a kiválasztott adatok a rekord elején vannak. Ezért célszerü az ellenőrzött alapfile-on a hivatkozás gyakoriságának sorrendjében elhelyezni az adatokat.

Az adatok többféle formában történő tárolása azért célszerü, mert a feldolgozás későbbi lépéseiben egyes kódolási igények gyakran fordulnak elő, igy ugyanazt a leképezést ismételten el kellene végezni, ha eleve nem gondoskodnánk az összevont kódértékek tárolásáról is. A kódolások ismételt elvégzése jelentős gépidő és tárolóhely kapacitást köthet le.

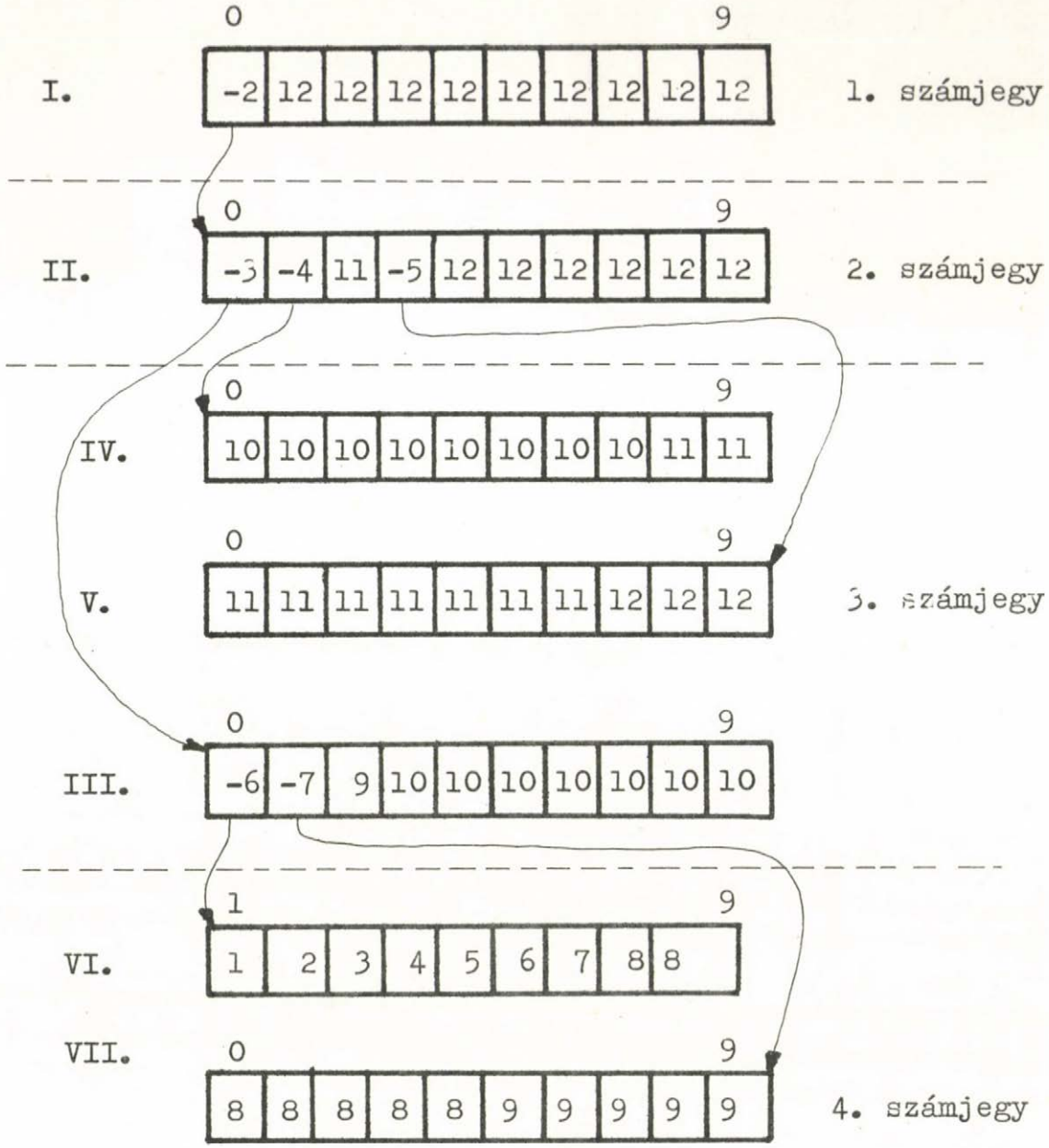
Tekintsünk egy példát. A morbiditási táblázatokban gyakran kell ápolási idő szerint bontott statisztikákat adni. Legyen mondjuk egy gyakran használt ápolási időcsoportositás a következő /6. táblázat/.

idő /nap/	1	2	3	4	5	6	7	8-14	15-29	30-179	180-369	370-
kód	1	2	3	4	5	6	7	8	9	10	11	12

6. táblázat

Adatok kódolására /igy például a fenti időcsoportok képzésére/ a táblafile készitésnél és a táblázatkiirásnál is van lehetőség, de mivel ezekben a részekben csak egyváltozós értéktáblázatok adhatók meg, ezért például a 6. táblázatban leirt függvény tárolásához 9999 rekesz szükséges - mivel ez az érték az ápolási idő legnagyobb kódolható értéke. Sok esetben ekkora szabad memóriaterület egyáltalán nem áll rendelkezésünkre.

Ha az előkészitő /ellenőrző, kódoló/ rendszer döntési gráfjait alkalmazzuk, és a négyjegyü ápolási nap kódot /mely 1 és 9999 között változhat/ négy egyjegyü /0-9/ adatként kezeljük, akkor ez a függvény 62 rekeszben is tárolható a 8. ábrán látható módon /a döntési gráfokról ld. a 3.1.2 pontot/.


0
9
I.
-2 12 12 12 12 12 12 12 12 12
1. számjegy
0
9
II.
-3 -4 11 -5 12 12 12 12 12 12
2. számjegy
0
9
IV.
10 10 10 10 10 10 10 10 11 11
0
9
V.
11 11 11 11 11 11 11 12 12 12
3. számjegy
0
9
III.
-6 -7 9 10 10 10 10 10 10 10
1
9
VI.
1 2 3 4 5 6 7 8 8
VII.
0
9
8 8 8 8 8 9 9 9 9 9
4. számjegy

8. ábra

4.1.3 Input-output rekordleirás

Az ellenőrző, kódoló eljárásban a felhasználó sorszám szerint hivatkozhat az input-output rekordelemekre. A rekordelemek mindig egész változók, tartalmuk értelmezése az input-output konverzió formátumától függ. A konverzió FORTRAN nyelven adható meg. Azokra a rekordelemekre, amelyeket nem kívánunk felhasználni X specifikációval, azokra amelyeket változatlan formában akarunk az output rekordba átvinni A specifikációval hivatkozhatunk.

Az ellenőrző, kódoló eljárás jobb áttekinthetőségének érdekében egy COBOL rekordleíráshoz hasonló adatleírást készíthetünk, feltüntetve mellette a kódolási eljárásokat, az új adatoknak az eredetiekhez viszonyitott elhelyezkedését /kódolt adatokat a már feleslegessé vált input elemek helyére is irhatunk/, az output adatokat és azok sorrendjét.

4.2 Válogatások és táblafile kombinációk

4.2.1 Válogatási taktika

A válogató /kódoló/ részrendszer /3.3 pont/ egyik célja a teljes adatrekord felbontása. Olyan

részrekordokat képezünk, amelyekből - táblafile-ok készitésén keresztül - többféle statisztikai táblázatot is készithetünk. Nyilvánvalóan az a célunk, hogy az igényelt statisztikai táblázatokat a lehető legkevesebb előkészitő lépés - pl. válogatás - után készitsük el. Általában minél több adat tartozik egy részrekordba, annál többféle táblázat készithető belőle. A rekordméretet a felesleges adatmozgatás elkerülése érdekében viszont jó minél kisebbre választani. A felhasználónak e két ellentmondó feltétel egyensúlyba hozásával kell egy optimális stratégiát kialakitania. A részrekordok összeállitásában segithet a következő egyszerü szervezési eszköz. Egy mátrixban ábrázolhatjuk a készitendő táblázatok és az adatok kapcsolatát. A mátrix oszlopai a feldolgozandó adatok $/A_1,A_2,\ldots,A_n/$, a sorok a táblázatok $/T_1,T_2,\ldots,T_m/$. Példaként tekintsünk egy kilenc sorból /táblázatból/ és hét oszlopból /adat/ álló mátrixot /7. táblázat/. A táblázatok és az adatok összetartozását X-el jelöljük. A mátrix sorainak, oszlopainak permutálásával és egyes oszlopok ismétlésével olyan $M_1,M_2,\ldots$ mátrixokat hozunk létre, amelyekben az oszlopok száma lehetőleg kicsi, miközben a sorok továbbra is egy-egy teljes táblázatot irnak le.

	A_1	A_2	A_3	A_4	A_5	A_6	A_7
T_1	X		X				X
T_2				X			
T_3		X			X		
T_4	X	X					
T_5				X		X	
T_6						X	
T_7			X			X	
T_8			X	X			
T_9	X				X		

7. táblázat

Ilyen átrendezett mátrixokat /részmátrixokat/ mutatunk be a 8. táblázatban. Ezek a mátrixok az $A_1,\ldots,A_7$

	A_3	A_4	A_6
T_2		X	
T_5		X	X
T_6			X
T_7	X		X
T_8	X	X	

M_1

	A_1	A_3	A_7
T_1	X	X	X

M_2

	A_1	A_2	A_5
T_3		X	X
T_9	X		X
T_4	X	X	

M_3

8. táblázat

adatok egy olyan szétválasztását mutatják be, amelyben /a 7. táblán adott/ $T_1,\ldots,T_9$ táblázatok előállithatók, miközben egyetlen részrekordban sincs háromnál több adat.

Természetesen az adatrendszer felbontására a fenti modell csak egy durva közelitést ad. Figyelembe kell még például azt is venni, hogy a gyakrabban használt részadatrendszerek terjedelmét célszerü kisebbre venni, tekintettel kell lenni arra is, hogy a táblázatok tartalmán tul egyéb szempontok is szerepet játszanak az adatrendszer felbontásában: például a részrekordok létrehozása mellett a rekordok között is szelektálni kell, egyes részrendszereken transzformációkat /rekordok összevonása, felbontása, rendezése/ kell végrehajtani. Az adatrendszer felbontása után a táblafileok készitése egy ujabb adatválogatást /a táblafile dimenzióinak kiválasztása/ jelent.

Ez a kérdéskör - matematikai vonatkozásait tekintve - a hipergráfok felbonthatósági problémáihoz kapcsolódik.

4.2.2 Táblafile-ok összeállitása

A táblafile-ok készitésénél a [13] tanulmányban illetve a 3.7 pontban felsorolt szempontok mellett most megvizsgáljuk a táblafile-ok méretének megválasztásával kapcsolatos problémákat.

A táblafile mérete a tábla dimenzióterjedelmeinek szorzata. Ez az érték viszonylag kis dimenziószám és dimenzióméretek mellett is nagyra nőhet. A SIS77 kifejezetten nagyméretü minták vizsgálatát szolgálja - mondjuk 100 ezres nagyságrendü /esetleg milliós/ mintákról lehet szó. Igy a tömöritett bináris file-ok is több százezer gépi szóból állnak. Ezek feldolgozása azonban lényegesen időigényesebb, mint egy hasonló méretü táblafile-é. Táblafile-ok megválasztásakor tehát felső korlátként akár a megfelelő tömöritett bináris file nagyságát meghaladó méretet is vehetünk, még akkor is, ha egy ilyen táblafile rengeteg üres poziciót tartalmazhat. Természetesen ilyen nagyméretü táblafile-okat csak részletekben /rendezett input file-ból, vagy az input file ismételt olvasásával/ készithetünk.

A táblafile-ok illetve a kinyomtatott táblázatok maximális dimenziószámának /8 illetve 4 dimenzió/ rögzitésekor egyrészt azt a szempontot vettük figyelembe, hogy egy statisztikai táblázatban egyszerre 3-4 változónál többet képtelenség áttekinteni, másrészt 8 változó, még ha csak 4-5 féle értéket is vesz fel, több tizezer illetve százezer értékkombinációt ad. Efölé az érték fölé lépni szervezés szempontjából

sem érdemes /kisebb táblafile-ok is sok táblakombináció kinyomtatását teszik lehetővé/, de egyébként is a jelenlegi gépkapacitások mellett nagyobb file-ok alkalmazása nehézkes.

I R O D A L O M

[1] KOMO'77 - Kórházi morbiditás adatinak számitógépes feldolgozása, Az egészségügyi információrendszer korszerüsitése /alapozó tanulmányok, rendszerleirások/ 4., ESZTIK /kézirat, belső használatra/, Budapest, 1977.

[2] Kralli A., Ratkó I., Ruda M., Soltész J., A statisztikai adatfeldolgozás matematikai és számitástechnikai problémái, MTA SZTAKI Tanulmányok, 70/1977.

[3] Nyiry G., Varga L-né, Statisztikai adatok feldolgozása MINSZK-2/22 számitógépen, INFELOR Közlemények, 3., Budapest, 1973.

[4] Ratkó I., Egy számitástechnikai eszköz bonyolult logikai kifejezések leirására orvosstatisztikai alkalmazásokban, Számitástechnikai és kibernetikai módszerek alkalmazása az orvostudományban és a biológiában, 8. Kollokvium, Szeged, 1977.

[5] Ratkó I., Bonyolult logikai kifejezések kiértékelésének számítástechnikai és optimalizálási problémái, MTA SZTAKI Közlemények, 20/1978.

[6] Ratkó I., On optimization problems of logical expressions in programming languages, Matematikai logika a programozáselméletben kollokvium, Salgótarján, 1978.

[7] Ratkó I., Döntések sztochasztikus optimalizációja adatfeldolgozásnál, VIII. Magyar Operációkutatási Konferencia, "Operációkutatás a gyakorlatban - 1978", Szeged.

[8] Ruda M., Egy általános információs rendszer kórházi morbiditási adatok feldolgozására, Számítástechnikai és kibernetikai módszerek alkalmazása az orvostudományban és a biológiaában, 8. Kollokvium, Szeged, 1977.

[9] Ruda M., Statistical Information System with Health Service Application, 4. Winterschool of Visegrád on the Theory of Operating System, Szentendre, 1978.

[10] Ruda M., Egy széles körben alkalmazható program-optimalizálási módszer, MTA SZTAKI Közlemények, 20/1978.

[11] Ruda M., Módszer a programkészités egyszerüsitésére, Számitástechnika, IX. évf. 7-8. sz., 1978.

[12] Gál A., Ruda M., Egy lehetőség Honeywell FORTRAN programok konverziós müveleteinek gyorsitására, SZÁMKI Tanulmányok, 1978/II.

[13] Ruda M., A SIS77 statisztikai információs rendszer kialakitásának szempontjai, alkalmazásának és továbbfejlesztésének lehetőségei, MTA SZTAKI Tanulmányok /megjelenőben/

[14] Ruda M., Optimalizálási kérdések a statisztikai adatfeldolgozásban, VIII. Magyar Operációkutatási Konferencia, "Operációkutatás a gyakorlatban - 1978", Szeged.

[15] Ruda M., Egy számitástechnikai módszer függvénytáblázatok tömör tárolására, egy adatfeldolgozási alkalmazással /kézirat/

[16] Soltész J., Egy általánosan használható kódolási eljárás és alkalmazása a hospitalizált morbiditási vizsgálatokban, Számitástechnikai és kibernetikai módszerek alkalmazása az orvostudományban és a biológiában, 9. Kollokvium, Szeged, 1978.

A TANULMÁNYOK sorozatban eddig megjelentek:

73/1978 S.A.COONS: Homogeneous coordinates, projective transformations, and conics

74/1978 WOLFGANG FRANKE: Vortäge über das Graphische Display GD'71

75/1978 VASKÖVI ISTVÁN-GALBAVY MÁRTA: Anyagszétválasztási rendszerek tervezésének és optimális üzemeltetésének általános megközelitése

76/1978 SOMLÓ JÁNOS-NAGY JUDIT: Módszer munkadarabok forgácsoló megmunkálási folyamatának optimalizálására

77/1978 SZÁSZNÉ, TURCHÁNYI PIROSKA: Optimalizálási feladatok csomagkapcsolt számitógéphálózatok tervezésénél

78/1978 DARVAS PÉTER - GALLAI ISTVÁN - HOSSZU PÉTER - KRAMMER GERGELY: Papers of Computer Graphics

79/1978 DR ADOLF KOTZAUER: Beschriftung und bemassung von automatisch erstellten zeichnungen unter benutzung des graphischen dialogs

80/1978 PRÉKOPA ANDRÁS: Studies in applied stochastic programming I. /Cikkgyüjtemény/

81/1978 PETER BONITZ: Ein Beitrag zur Theorie des Entwurfs doppelt gekrümmter Flächen unter differentialgeometrischen und rechentechnischen Aspekten

82/1978 TANKÓ JÓZSEF: Szabályos job-folyam párok ütemezésének vizsgálata I.

82/1978 TANKÓ JÓZSEF: Szabályos job-folyam párok ütemezésének vizsgálata I.

83/1978 TANKÓ JÓZSEF: Szabályos job-folyam párok ütemezésének vizsgálata II.

84/1978 BÁNYÁSZ CSILLA - KEVICZKY LÁSZLÓ: Discrete time identification of linear dynamic process

85/1978 DR HOFFMANN PÉTER: Számitógépes szerszámgépvezérlés egy alkatrészprogramozási módszere

86/1978 RUDA MIHÁLY: A SIS77 Statisztikai Információs Rendszer kialakitásának szempontjai, alkalmazásának és továbbfejlesztésének lehetőségei

87/1978 Téli iskola

88/1978 GAÁL BALÁZS - HERMANN GYULA - HORVÁTH LÁSZLÓ - RENNER GÁBOR - VÁRADY TAMÁS: Szoborszerü felületek tervezése és megmunkálása