MTA Számitástechnikai és Automatizálási Kutató Intézete, Közlemények 18/1977.

SZÁMITÓGÉP RENDSZEREK TERVEZÉSE ÉS SZIMULÁCIÓS MODELLEZÉSE

Stauder Ernő
KSH Számitástechnikai Igazgatóság

1. A feladat megfogalmazása

A számitástechniaki szolgáltatásokat felhasználók gyakran teszik fel a kérdést, hogy miért tart olyan sokáig egy uj számitóközpont létrehozása, vagy egy már müködő központ jelentős bővitése. Kérdésük jogos, legalább is a felhasználó természetes kiváncsisága oldaláról. Hosszú ideje foglalkozik ezzel a kérdéssel a számitástechnikai szakemberek egy köre, akik részben vagy teljesen felelősek a számitóközpontok felállitásáért, üzemeltetéséért, vagy éppen annak vezetéséért. A hatvanas évek elején egy közepes számitóközpont létszáma 100-150 fő körül mozgott. A különböző szakosodások és a gépek teljesitményének növekedése eredményezte, hogy a hetvenes évek elejére ez a létszám közel kétszeresére nőtt mindenütt. A szakemberek egy része ezt természetesnek veszi, mert a korábbi évek "programozó centrikus" szervezeti felépitését – és amiből ez adódott, a feladatok ilyen értelmü tervezését – felváltotta egy igen sokszintü munkamegosztáson alapuló heterogén rendszer. A számitóközpontok, mint müködő rendszerek, a funkciójukat tekintve különböző feladatokat látnak el.

a.) **Szolgáltató központ**

A felhasználó a kivánt formában, az általa összegyüjtött és valamilyen módon rögzitett adatokból *feldolgozást* kér (és remélhetőleg kap). A felhasználók köre lehet véges, meghatározott – ekkor adott feladatok végrehajtására létrehozott számitóközpontról beszélünk. (tipikusak a termelő vállalatok saját számitóközpontjai). Ha a felhasználók körét nem korlátozzuk, ugy *bérmunka iroda* jellegü számitóközpontról beszélünk. Mindkét esetben jellemző, hogy a külső felhasználók szemszögéből az SzK egy *szolgáltató*, speciális feladatok elvégzését vállaló szervezet.

b.) **Termelő üzem**

A számitóközpontokban dolgozók ma már egyre jobban felismerik és tapasztalják, hogy a munkájuk milyen mértékben specializált és mennyire illeszkednie kell *időben és térben* a többi kapcsoló munkafolyamathoz. Bátran kimodhatjuk, hogy egy számitóközpont, saját müködését tekintve, egy üzemhez vagy gyárhoz hasonlitható. Speciális "terméket" állit elő, információt, azokból az adatokból, amelyet a megrendelő előirt számára. Még jobban közelithetjük a fizikai valóságot, ha azt mondjuk, hogy a megrendelő által rendelkezésre bocsátott vagy korábbi feldolgozásból származó és megőrzött adatok (megfelelő formában rögzitett) felhasználásával uj adatokat, illetve a korábbi adatok ujabb rendezett formáját állitja elő a számitóközpont. A "termelő folyamat" során a technológiát a feladat *szervezése* irja elő, az alkalmazott technológia pedig a rendelkezésre álló *software* függvénye. A felhasznált munkaerő tipusát a részfela-

dat jellege szabja meg: szervező, programozó, operátor, adatrögzitő, stb. A felhasznált eszköz pedig legtöbbször az SzK valamelyik *hardware* berendezése, vagy egyidejüleg több berendezése. Természetesen az ilyen "üzem" számos segédfolyamatot is igényel a "termelés" zavartalanságának biztositására, és ezeket ténylegesen (vagy ujtipusu fedő nevek alatt) meg is lehet találni minden számitóközpontban.

c.) Információs és adatraktár

A termelő üzem jellegnek a tárgyalásánál utaltunk a korábbi feldolgozásokból nyert adatok ismételt feldolgozására. Erre lehetőség nyilik azáltal, hogy az adatokat megfelelő tároló közeg – papir vagy mágneses alapanyagu – felhasználásával hosszú időre *tárolhatjuk*, megőrizhetjük. Az igy tárolt adatok gépi uton ismét beolvashatók. Ezek az adatok meghatározott felhasználóknak információkat jelentenek. Éppen ezért az adatok hosszú időre való tárolása egy speciális *raktározási* feladatot jelent a SzK számára. Ennek fontosságát már egyre jobban hangsulyozzák, különösen az adatok biztonságának kérdését. Ez elsősorban a meghatározott célra létrehozott számitóközpontok esetében igaz, de esetenként a bérmunkát végző SzK is elláthat ilyen feladatot, az ügyfelek egy körének.

Az előzőekben tulajdonképpen mindig ugyanarról a számitóközpontról beszéltünk, de különböző nézőpontról. Vizsgálatainakt a *müködő számitóközpontnak*, mint termelő egységnek az elemzésére koncentráljuk. Ezen belül is feltételezzük a *software* – technológia – adottságának és olyan modellezési módszereket keresünk, amelyek a számitóközpont terhelésének növekedésével összhangban biztositani képesek az erőforrás szükségletek megbizható becslését.

Különösen fontos a számitógép konfigurációjának megfelelő méretezése, mivel ez a legdrágább berendezés az összes között. Ezt egyfelől a várható munkák terhelése alapján méretezhetjük, másfelől az egyes konfiguráció elemek paraméterei alapján becsülhetjük az átbocsátóképességet. Érdekes lehet a SzK szellemi kapacitásának a becslése, teljesitmény és átbocsátóképesség alapján. A következőkben ezekre a kérdésekre igyekszünk választ adni.

2. A modellezés, mint tervezési módszer

Akár egy már üzemelő számitóközpont kapacitás bővitéséről, akár egy uj számitóközpontban üzemeltetendő uj berendezés méretezéséről beszélünk, minden esetben szükséges az elvégzendő feladatok lehető legpontosabb becslése. Ez a gyakorlatban legtöbbször a feldolgozások funkcionális egységeire (munkafolyamatok, munkalépések – azaz programok és programrendszerek) specifikusan értelmezett erőforrás szükségletek (CPU és) vagy start-stop idő, periféria igény) megadását jelenti. Természetesen ez a megadás már magában hordozza annak a konfigurációnak *specifikus* paramétereit, amelyet elképzelünk, vagy amelyen mértünk. A tervezési munka pontos végzésénél ezt mérlegelni kell) amit bizony sokszor "elfelejtünk"), és ennek egyik biztositéka lehet az adaptiv tervezési módszer. Ez a konfiguráció

azt jelenti, hogy a rendszer-független paramétereket igyekszünk kihangsulyozni és alapul venni, és az egyes méretezési részeredményeket az ujabb rendszerek (variánsok) kimunkálásánál már felhasználjuk. Természetesen továbbra is számottevő bizonytalansági tényező lehet a tervezés időpontjában a jövő – azaz esetleg a néhány év mulva értelmezett – számitógép rendszerének munkaterhelésének becslése. Ennek értelmezéséről, a méretezési feladat technikai kivitelezésében való befolyásáról itt nem kivánunk beszélni, mivel ez a módszertanról elterelné figyelmünket. A terhelést megadó feldolgozási paraméterek fentiekben megadott finomságu és léptékü ismerete azonban teszi a feladat tárgyát képező rendszer elég pontos lebontását. A számitógépet, mint rendszert olyan rendszerelemekből felépülő egésznek képezzük le, amelyhez illeszteni lehet a rendszer feladatát jelentő adatfeldolgozási munkákat, azok specifikus paramétereivel. Ez lehetővé teszi a teljes feldolgozási rendszer dinamikus modellezését, és a modell megfelelő müködtetése révén a méretezéshez szükséges rendszerparaméterek mérését. A becsült vagy kivánt paraméterek elérésénél rögzithetők a feldolgozási rendszerre jellemző alapértékek és ezeket tekinthetjük a számitógép konfigurációt leiró értékeknek.

A számitógép konfigurációnak az ismertetett módszertanon alapuló meghatározása elvezet a *makro-szintű rendszer modellezéséhez.* Ez várhatóan pontosabb tervezési horizontot tud majd biztositani, mint amit a hagyományos módszerek (általában elég sok és jelentősen torzitó becslést felhasználva) igérnek. A modellezés módszertanának bemutatására felhasználjuk a Gaver által kidolgozott makro-modellt [2], amelyt Hansmann és munkatársai módositott formában a gyakorlatban is felhasználtak rendszermérésre és méretezésre [3]. Hangsulyozni kell, hogy a modell alkalmazhatósága nem korlátozodik csak hosszutávu tervezésre, konfiguráció meghatározására, az alkalmazható rövid távu kapacitás tervezésre is. Nem látszik elég jónak speciális alkalmazási feltételek mellett, például az adatbázis orientált feldolgozások esetében. Erre más modellek állnak rendelkezésre. [1, 5] Természetesen ezek a modellek elméleti modellek, érvényességüket az adott lehetőségek mellett ellenőrizni kell, mielőtt elfogadnánk a tervezés módszertanaként.

A számitógép müködését tekintve megengedhető olyan egyszerűsités, amely az 1. ábrán látható strukturális elemekre bontja fel a rendszert. Ezután a feldolgozásra kerülő munkák, programok ilyen részekhez való hozzárendelése, illetve ezekből az erőforrásokból való igénylése szempontjából vizsgáljuk most ezt a modellt. A számitógép terhelését tekintve alkalmazzuk azt a megközelitést, hogy a terhelést a mindenkor a memóriában (tárolóban) található "program szegmensek" halmaza adja. Ezek a szegmensek minden esetben három különböző müveleten mennek keresztül: beolvassa az adatokat (input) a tároló adott részébe (partició, szegmens, stb.), bizonyos CPU időt használt fel a feldolgozásra, majd az eredményeket közli (output).
A modellben nem kerülnek megkülönböztetésre az alkalmazások vagy programok különböző tipusai. A terhelést kizárólag az I/0 és a CPU időráforditásokkal mérjük, minden egyes szegmensre.

A modellben feltételezzük a következőket:

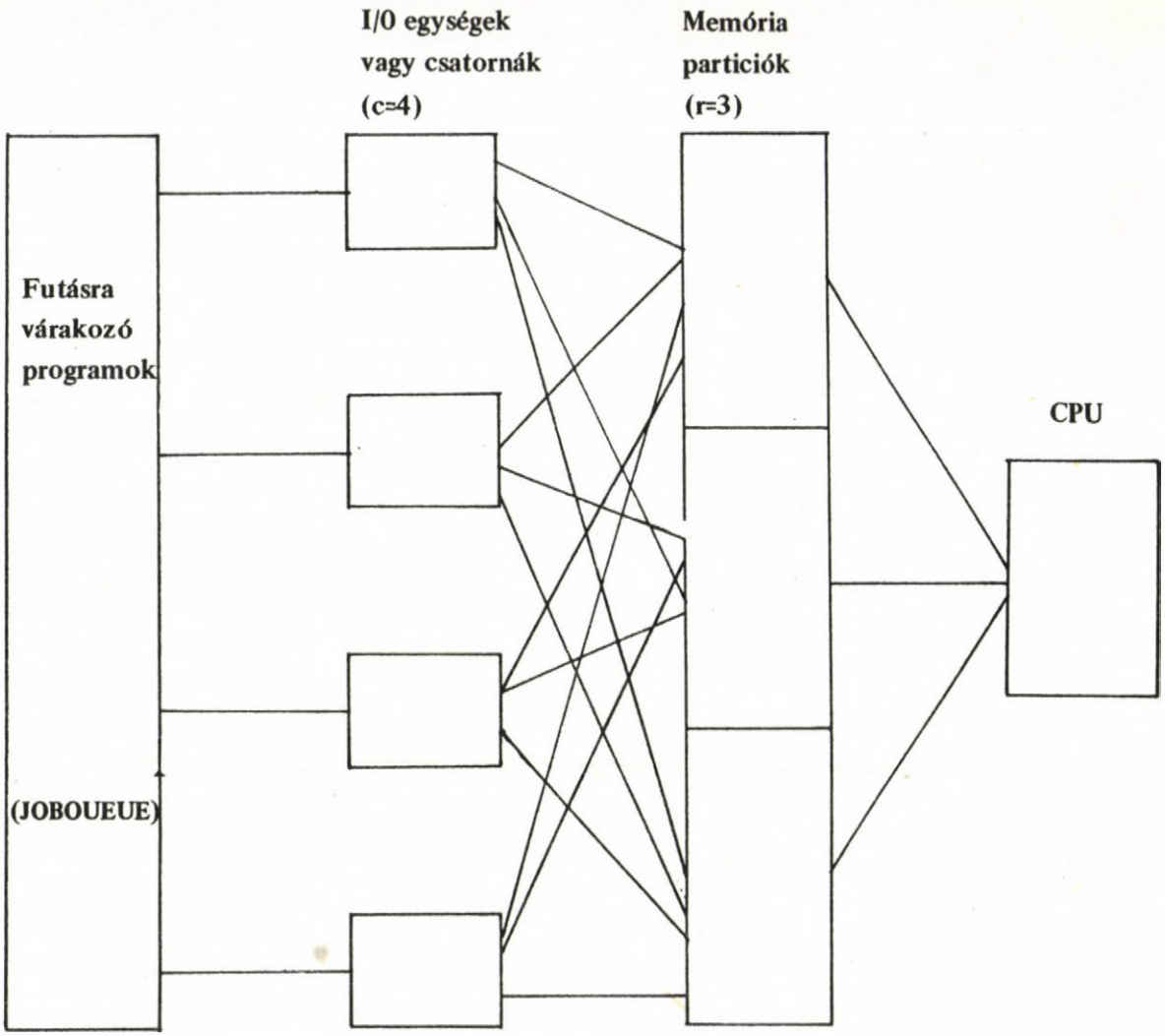
– A munkát, programokat a rendszer szegmensről szegmensre haladva dolgozza fel (hagyományos "lépésről-lépésre" módon);

- Minden szegmens egyetlen tároló terület (partició, stb.) és egyetlen I/0 egységet, illetve csatornát igényel;
- A felhasznált CPU idő szegmensenként változik, és az csak a véletlentől függ, az eloszlásról pedig feltételezzük, hogy exponenciális (a);
- A szegmensként felhasznált I/0 időről feltételezzük, hogy véletlen változó és eloszlásra jellemző, hogy exponenciális (b);
- A feldolgozás folyamatos, állandóan van feldolgozásra váró program;
- A központi tárolóban az egyes szegmensek által elfoglalt tárolóban az egyes szegmensek által elfoglalt tároló terület állandó nagyságu, és rögzitett. Igy a particiók számának növelése egyben a teljes memória kapacitás növelését is jelenti.

A fenti megszoritások eredménye az, hogy miután egy partícióhoz egyetlen csatorna (periféria) használatot feltételezünk, nincs várakozási idő addig, amig a csatornák (c) száma egyenlő vagy nagyobb a particiók (r) számánál. Továbbá az állandó munkaadagolás biztositja, hogy a csatornák felszabadulásával a partició ismét munkaképes lesz. Miután ezek a közelitések a valóságos feldolgozási rendszerkörnyezethez képest, korrekciós tényezők alkalmazásával pontosithatók a méretezési összefüggések.

A terhelés meghatározásához először vezessük be a a *foglaltság* fogalmát (f). Bár a szegmensek álatl felhasznált CPU és I/0 valószinüségi változó, mintavételek segeitségével meghatározható a CPU átlagos terhelése. A CPU foglaltsága befejeződik, ha a CPU nem talál ujabb olyan particiót, amelyre I/0 müvelet befejeződött. Eztuán a CPU *várakozik* (w). Ezeknek a felhasználásával kifejezhető a CPU átlagos terhelése.

(1) $$\overline{p} = \frac{\overline{f}}{\overline{f} + \overline{w}}$$



1. ábra: A Gaver modell szerinti struktura

A terhelésnek egy másik jellemzője lehet az átlagos számitási intenzitás. Ez a szegmenseknél CPU és I/0 idő arányát tükrözi. A [2] összefüggésben t_c és t_u valószinüségi változok, átlagértékeikkel származtatható az átlagos számitási intenzitás.

(2) $$\overline{\lambda} = \frac{\overline{t_c}}{\overline{t_u}}$$

Feltételezve a szegmensekben végzett munkák egymástól való függetlenséget, valamint exponenciális eloszlás szerinti viselkedést, igy a CPU átlagos terhelése csak a particiók és a csatornák számától függ, a következő összefüggés szerint:

(3) $$p = g(\overline{\lambda} \mid r, c)$$

Ebből tovább levezethetjük az egy particióra vetitett átlagos terhelés értékét:

(4) $$\overline{p}' = \frac{\overline{p}}{r}$$

Általánosságban értelmezhető az átlagos csatorna terhelés is, ha a CPU terheléssel közvetlen összefüggését elfogadjuk:

(5) $$q = \frac{\overline{p}}{\overline{\lambda} \cdot c}$$

Már itt szükséges és időszerü bizonyos korrekciós tényező bevezetése. Ugyanis a particiónak az állandó terhelést nehezen lehet biztositani, ezért állásidőre kerül sor a particiókon belül. (Vigyázzunk, ez nem azonos sem mérésben, sem jellegben azzal a várakozási idővel, amelyet a partícióban helyet foglaló program szegmens tölt el a következő I/0 müvelet befejezéséig!.) Ezért a modellben alkalmazott névleges partició számot (r), helyes egy effektiv partició számmal (r_e) figyelembe venni, ugyanakkor az egyes particiók is egynél több csatornát, illetve I/0 egységet igényelnek, igy erre is célszerü bevezetni egy effektiv értéket. Ezt egy korrekciós tényező bevezetésével érhetjük el az átlagos számitási intenzitás összefüggésében, és mondjuk azt, hogy az uj átlagérték legyen $\gamma\overline{\lambda}$. Ezzel a (3) egyenlet a következőképpen módosul:

(6) $$p = g(\gamma\overline{\lambda} \mid r_e, c)$$

Ez a modell most már alkalmasnak látszik a konfiguráció és a terhelés kölcsönös tervezésére, mind hosszutávu, mind rövidtávu becslések esetében.

3. A modell alkalmazása

Egy alkalmas kifejlesztett modell értékének, a gyakorlati alkalmazhatóságának ellenőrzésére Hansmann és munkatársai [4] számos mérést és elméleti számitást végeztek, különböző számitógép konfigurációk esetében. Gyakorlati mérések gerincét egy IBM 360/65 konfiguráción nyert adatok képezték, ahol,igen részletes adatgyüjtést végeztek a napi feldolgozásokról egy 16 órás termelési periodus során. A következő adatokat mérték:

- Partició fogalaltsági idő, a start-stop időkülönbség alapján;
- Program tipusok eloszlása;
- CPU produktiv idő az egyes szegmensek feldolgozásakor (az időegység 10^{-4} óra volt);
- CPU idő programonként;
- I/0 idő szegmensenként (ezt nem közvetlen méréssel nyerték);
- I/0 idő megoszlás a különböző periféria (csatorna) tipusokra (szintén közvetett méréssel nyerték).

Ezeknek az adatoknak a felhasználásával a számitási intenzitást, az effektiv partició számot és a CPU átlagos terhelését számitották, elsősorben a rövididejü szegmensek esetére. Ezeknek az adatoknak az alapján megkülönböztethető volt különböző partició vagy szegmens élettartamu feldolgozások, csoportja, különböző számu aktiv szegmens egy adott időben és ezek eloszlása, valamint az ezekhez tartozó I/0 idő felhasználás. Anélkül, hogy e részletes elemzésbe belemennénk, amely részben abból az alkalmazásból, részben az olyan tipusu adatgyüjtésből kinálkozik, néhány általános következtetést érdemes kihangsulyozni.

Adott számitóközpont jelenlegi terhelésének mérésére ma már igen sokféle részletes adatot tudunk gyüjteni, részben hardware, részben software mérési módszerekkel. A folyamatos mérés és adatgyüjtés lehetővé teszi a számitóközpont, illetve a konfiguráció terhelésének ellenőrzését, esetenként a terhelésének szabályozását. Az üzemeltetésről gyüjtött részletes adatok [8] esetenként közvetlenül felhasználahtók a várható terhelés becslésére, máskor pedig összetettebb statisztikai elemzések után alkalmazhatók, a korábban ismertetett modellnél input adatként. A rendszertervezési munkához a már müködő rendszerben mért adatok igen nagy segitségét jelentenek, mivel ezek támpontot szolgáltatnak egy uj konfiguráció különböző alternativáinak értékeléséhez.

Ezen az uton kiválaszthatók az elfogadható vagy elvetendő konfigurációs megoldások. A számitásokat a Gaver modell alapján végezhetjük, az ehhez szükséges adatokat pedig adott software monitor, például az IBM SMF/ (System Management Facility), alkalmazásával gyüjthetjük. Ezek után a módszer alkalmazható mind rövidtávu, mind hosszutávu tervezésre.

A rövidtávu terhelés tervezésénél könnyü dolgunk van, mert a várható struktura a módositás időpontjában valószinüleg hasonlitani fog a mérési időpontban tapasztalt program és feldolgozási strukturához. Igy számos konfiguráció lehetőségre alkalmazzuk a Gaver modellt és utána rangsoroljuk azokat, valamilyen kiválasztási politika szerint. Itt természetesen most már szállitási határidők, költségtényezők és egyéb, a számitóközpont szempontjából lényeges tényezőt is figyelembe kell venni.

Nem ilyen egyszerü az eset a *hosszutávu* tervezés esetében. Nagyon sokan kétségbe vonják egyáltalába hosszutávu tervezés létjogosultságát, különösen ilyen esetekben. A számitástechnika rohamos terjedése, szerepének vitathatatlan növekedése azonban nem engedi meg a polemizálást erről a témáról, hanem sürgeti az alkalmazható módszer bevezetését. Ezen a pon-

ton ismét hangsulyozni kell a modellezés fontosságát, mind módszertanilag, mint tervezéstechnikailag.

3.1. Az átbocsátóképesség vizsgálata

A SZK müködések egyik mércéje, hogy milyen volumenü feldolgozást tud valamilyen időegységre vetitve elvégezni. Mint *termelő üzem* a számitóközpont hasonlóan vizsgálható más termelő tevékenységet folytató vállalathoz. Azonban a termelés a folyamatában, az adatfeldolgozó teljes rendszerében igen különböző tipusu erőforrások kerülnek felhasználásra. Az egyensuly, vagy az optimum megtatálása nem minden esetben egyértelmü, különösen a költségek nem homogén jellege miatt. A tapasztalat azt mutatja, hogy a számitóközpontok nem alkalmaznak egységes elszámolási rendszert, ez pedig még mindig a különleges helyzetükből, a kinálatot meghaladó kereslet adta piaci helyzetből származik. Minden esetre a SZK szempontjából biztosan célszerü a minél több munka vállalása és remélhetőleg teljesitése. A terhelés maximális szinten tartása viszont megfelelő tartalék munkát, előre vállalásokat igényel. Ugyanakkor ezekre is reális átfutási időket kell meghatározni, hogy elfogadható teljesitési határidőt tudjunk adnia a megrendelőnek.

A teljes feldolgozási folyamatban az egyik kritikus pont maga a számitógép, amely a mai "harmadik generációs" világban a multiprogramozási környezetével igen sok lehetőséget kinál. Mint azonban már a 2. pontban utaltunk rá, igen nehéz bizonyos paraméterek becslése a feldolgozáshoz szükséges pontos átfutási idő meghatározásához, éppen az alkalmazott és software bonyolult müködésének részletes nyomonkövetése miatt. A matematikai és az operációkutatás területén kidolgozott uj módszerek, a valószinüségszámitás és a sorbanállási elmélet legujabb kutatási eredményei azonban biztató lehetőséget kinálnak a problémának a megoldására is. A számitógépek, aSZK átbocsátóképességének, a feldolgozások átfutási idejének becslésére már a 60-as évek közepén születtek modellek, szimulációra alapozott elemzések [6, 9, 17]. Ezek a batch-tipusu feldolgozási környezetet feltételezték alapvetően. Egy egészen speciális, de egyre jobban terjedő üzemeltetési környzet, a time-sharing (idő-osztásos) rendszerben történő feldolgozások paramétereinek becslésére megint más modelleket kellett kidolgozni [10, 11, 12]. Végül a 70-es évekre eljutottunk oda, hogy speciális programcsomagokat, illetve "nyelveket" fejlesztettek ki különböző számitógéprendszerek paraméteres elemzésére. [13, 15, 16]. Mivel a vizsgálat tárgyát képező rendszer önmagában igen bonyolult, következésképpen mindegyik rendszer valamilyen mértékü elhanyagolással él, makro szinten igyekszik lehetőséget adni a rendszer-elemzőnek paraméterei meghatározására.

Minden számitógéprendszer modellezése azonban önmagában hordja a modellező azon törekvését, hogy megszabja a szükséges információk körét és a vizsgálat szempontjából elégséges részletezettségét. Az "ideális" modellt tehát kettősség jellemzi:

a.) Képes a hardware berendezések széles skáláját és az operációs rendszerbelső logikáját együtt szimulálni, azok paramétereinek dinamikus változtatásával, a modell ujradefiniálása nélkül.

b.) Lehetővé teszi a számitógéprendszer különböző részeinek különböző részletességü definiálását, a feladat által megkivántak szerint.

A legtöbbször nehéz a modellt igy felépiteni, és esetenként az elsősorban az ésszerü egyszerüsitések miatt adódik. Kisebb szerepe van a modellezésben használt programnyelv, vagy a már meglévő rendszer adta korlátozó tényezőnek, de ezt is meg kell emliteni.

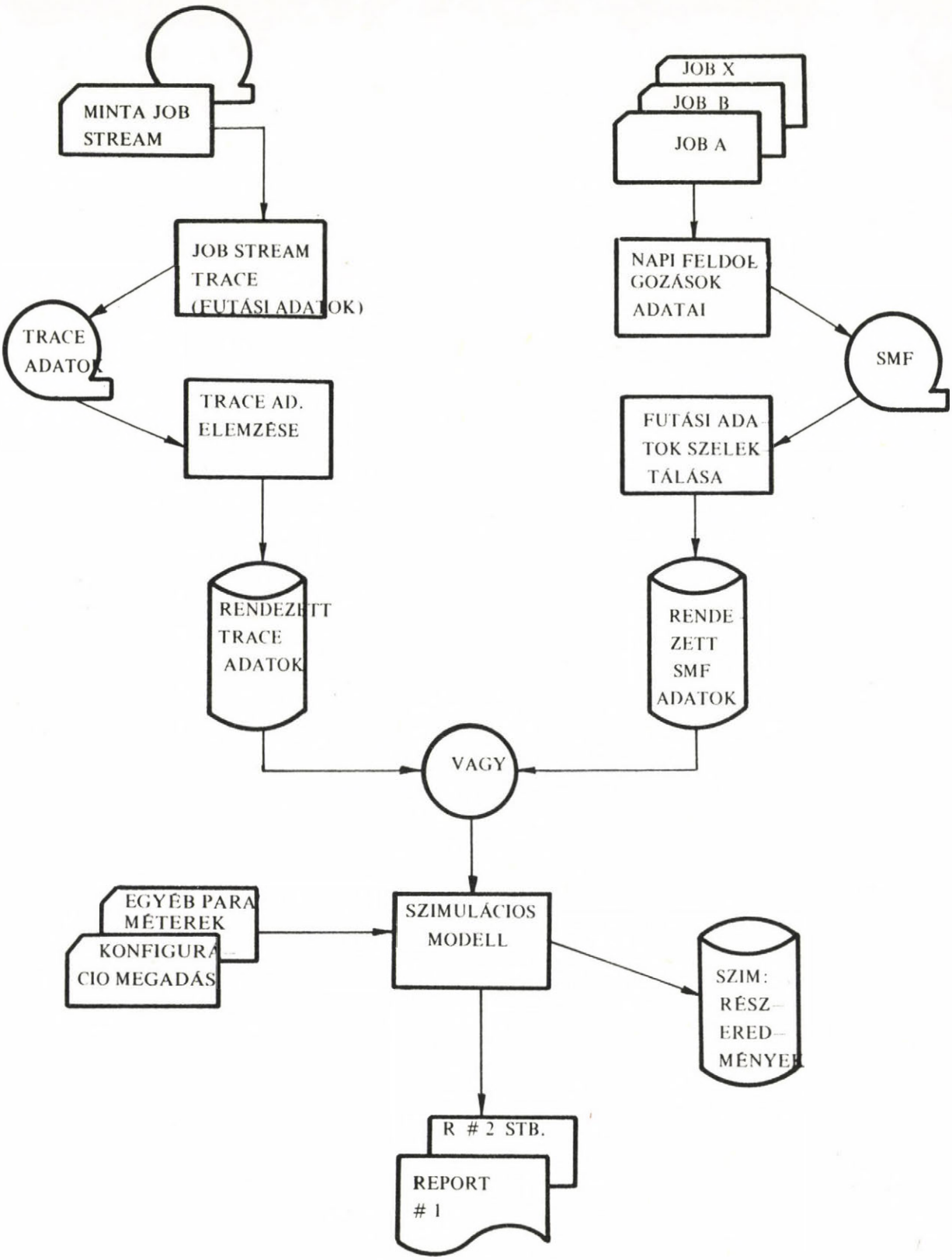
Abban az esetben, ha egy meglévő számitógép konfigurációja által átbocsátható programok számáról és tipusáról akarunk meggyőződni, megfelelő statisztikát kell készitenünk egy-egy periodus során az átbocsátott programokról és a terhelési struktura változtatásával elemezhetjük, hogy a különböző munka összeállitások esetében hogyan alakulnak a rendszer terhelési és az átbocsátott munkák időparaméterei. Ez a tipusu elemzés azonban nem tipikus.

Gyakoribb, talán a legtipikusabb vizsgálat az, amikor elég jól ismert a várható terhelés és kiváncsiak vagyunk, hogyan alakulnak a komplex rendszer különböző pontjain a mért értékek, ha a rendszer funkcionális összetevőit változtatjuk.

Miután a számitógép konfiguráció müködése jól meghatározott, annak belső változtatása hatással van egyéb rendszer pontokra is. Éppen ezek a hatások tisztázódnak, amelyre a rendszer bonyolultságából adódóan csak *teljeskörü* vizsgálat adhat választ. Ennek módszertana a modellezés, technikája a számitógépes szimuláció. Az ilyen rendszervizsgálatnál kétféle adatot kell inputként értelmezni.

Az első a vizsgált konfiguráció leiró adatok halmaza, külön-külön értelmezve az egyes modell vizsgálatokra. A másik a rendszer terhelését jelentő munkák, programok jellemző adatai. A konfigurációt leiró adatok általában statikusnak tekinthetők, néhány speciális esetben azonban szokás a tényleges paraméter értékét valószinüségi változónak tekinteni (pl. adott pillanatban a feldolgozási program által megcimzett sáv száma a mágneslemezen). A feldolgozási adatokat átlagértékként kezelhetjük, származtatásukra ismét a monitor által gyüjtött adatok, az abból képzett adatbázis szolgálhat. (Természetesen, ha még nincs számitógépünk, ugy ezeket az adatokat csak becsülni tudjuk. Lehetséges próba futtatások, próba üzemek végzése is hasonló konfiguráción, hogy legalább reális alapok legyenek a becslésre). A monitor által gyüjtött elemi adatok a következő csoportba sorolhatók:

- program betöltés (program inditás ideje;
- program befejezés ideje;
- állomány megnyitása (OPEN);
- állomány lezárása (CLOSE);
- csatorna (periféria kérés (I)0 tevékenység, tipus szerint);
- várakozás (WAIT);
- I/0 befejezésre várakozás /I/0 (INTERRUPT).



2. ábra A szimulációs modellezés folyamata

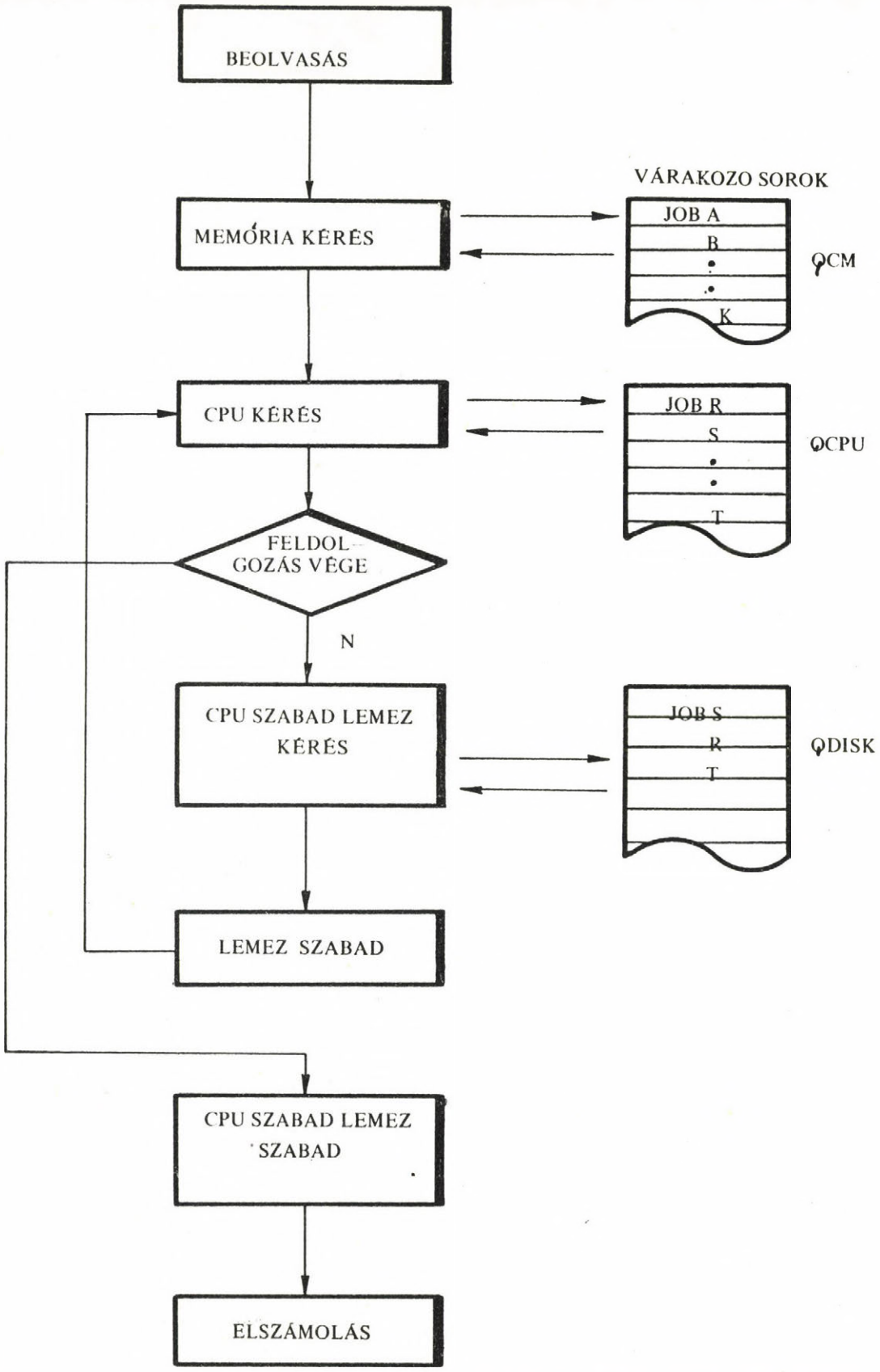
Ezeket az adatokat meghatározott időperiodus során mérjük, és rendezésükkel összeállítható egy tipikus terhelési struktura a vizsgált konfiguráció input job-folyamatként. A 2. ábra a modellezés általános folyamatát mutatja, az előbbi gondolatmenet alapján.

A szimulációs modellből egyrészt mérési eredményeket gyüjtünk a különböző konfigurációs pontokról, másrészt igyekszünk ujabb statisztikát felálltani az egyes munkák várható átfutási, várakozási és feldolgozási idejére. Ezután a modell változtatása nélkül, annak paramétereit vagy a terhelést változtatjuk – több munkát "pumpálva" a rendszerbe – és ismét gyüjtjük az eredményeket. Azok értékelése után, megfelelően rendszerezett formában már felhasználhatók közvetlenül a döntéshozatalhoz. Ez vagy a konfigurációra, vagy a vállat feladatokra összpontositva jelentkezik, de biztos, hogy a SZK optimális terhelésének irányába kell, hogy mutasson.

3.2. A számitógép konfiguráció komplex modellje: többcsatornás kiszolgálási rendszer

A 2. és 3.1 pontban egyaránt utalunk arra, hogy a számitógéprendszerek mai architektúrájának tekintve, sorbaállási rendszerként tekinthetők. Ezen az alapon kiindulva, a matematikusok először egyenletek és valószinüségi összefüggések sorozatával igyekeztek választ adni az ilyen rendszerek különböző viselkedésére. Bár a matematikai megfogalmazásoknál igen jelentős egyszerüsitéseket is alkalmaztak, illetve feltételeztek a rendszer viselkedéséről, számos összefüggés jól használható ezek közül, a különböző méretezéseknél [18]. A rendszerelemző azonban nem elégszik meg az átlagértékkel szereti tudni azt is, mi van az átlag mögött, melyik a szélső értékek. Erre a választ csak olyan dinamikus modellek felépitésével kaphatjuk meg, amelyek önmagukban hordozzák a sorbanállási rendszerek minden tulajdonságát. Ezt az elvet követjük akkor is, amikor a 3.1. pontban leirt modellezési technikát a gyakorlatban alkalmazzuk, ahogyan ezt az egyik kisérleti tervezésnél követték is [19]. A modell elméleti gyökere korábban vezethető vissza, [1], és a feldolgozást a következő elemi lépésekre bontja fel;

1. Program beolvasása a rendszerbe (várakozik szabad memóriára).
2. Amennyibe van szabad memória, betöltik a programot (t.i. az operációs rendszer).
3. A program feldolgozást (CPU) kér, ha nincs más program feldolgozás alatt. A CPU foglaltság a következő I/0 igénylésig tart.
4. Az I/0 kérésnél felszabaditja a CPU-t más munka számára. Ugyanakkor sorbaáll az I/0 teljesitéséért. Ha a sor üres – más program nem kérte ugyanazt a tipusu I/0-t – azonnal kezdetét veszi a kérés teljesitése.
5. Amint az I/0 befejeződik, a periféria (csatorna) szabaddá válik más program (I/0 kérés) számára. Ezután ismét a CPU kérésre kerül sor.
6. Amint a program befejezést nyer, felszabaditja a tároló területet egy következő munka számára.
7. A program elhagyja a rendszert.



3. ábra Egy program futásának belső feldolgozási lépései

A fenti lebontás természetesn sok egyszerüsitést is tartalmaz, de elég részletes ahhoz, hogy funkciónálisan a teljes számitógéprendszert vizsgálni tudjuk a feldolgozás oldaláról. A 3-4-5 folyamat annyiszor ismétlődik, ahány I/0 feldolgozási ciklus tartozik egy programhoz. A program feldolgozási lépései a 3. ábrán jól láthatók.

A feldolgozás legkisebb mért egysége a program (job), igy további részekre bontható, és ezek önmagukban "önálló életet" kezdenek élni, természetesen továbbra sem feledkezve meg "hovatartozásukról". Ugyanakkor uj kapcsolataik alakulnak azáltal, hogy más programok hasonló elemi lépései ugyanazt az erőforrást igénylik, azonos időben. Ekkor döntést kell hozni, "magasabb szinten", hogy melyik igénylő nyer kielégitést az erőforrással való kiszolgáláskor. Ez is jól látható a 3. ábrából, egyszerüsitett formában. Itt csak jelezhetjük a lehetséges várakozási sorokat: a memória, a CPU, a lemez, mint a leggyakrabban használt erőforrások, a kiszolgálásra várók sorából "táplálkoznak". Ahány tipusu erőforrást tudunk megkülönböztetni a rendszerben – és természetesen ezekre kiszolgálást kérni a programokból – annyi lesz legalább a rendszerben kezelt sorok száma. Tovább finomitható a rendszer általaz, hogy a perifériákat tipus szerint csoportositjuk, majd ezeket csatornákhoz rendeljük – ahogyan az a tényleges kofigurációkban is szerepel. Mivel egy csatornára többféle periféria (vagy csoport) köthető, esetenként ugyanaz a periféria (vagy csoport) több hardware csatornához is csatlakozhat, máris eljutottunk egy olyan bonyolult sorbaállási rendszerhez, amelynek vizsgálatára már semmiféle analitikus, matematikai formulákkal kezelhető módszer nem járható. Ehhez járul még a multiprogramozási környezetből adódó rendezett, dinamikus strukturált erőforrás kérések halmaza, amelyek együttes kezelése és értelmezése jelenti az *igazi* feladatot a modellezőnek, a szimulációit végzőnek. Látni kell azonban azt is, hogy egy ilyen modellezés igen költséges, még akkor is, ha már igen gyors gépek állnak rendelkezésre. (Érdekesség kedvéért megemlitendő, hogy az IBM erre a célra kifejlesztett programrendszere, s CSS [13] az első változatában a 360/40 gépen húszszor annyi időt használt fel, mint a szimulált periódus és 15 microsekundum periódus szimulálására a nagyon gyors 360/65 gépen is 0.24 millisekundumra volt szükség!). Éppen ezért nagyon oda kell figyelni, hogy a tervezés szempontjából milyen mértékig érdemes finomitani a rendszert – a tervezési költségek ésszerü korlátozása érdekében. Nem kétséges azonban, hogy már a számitógéprendszerek konfigurációs tervezését, a megfelelő feldolgozási követelmények pontos kielégitése érdekében, nem lehet csak egyszerü másolással, tipikus konfiguráció összeállitással elintézni. A SZK igen költséges üzem – mind önmagában, mind a felhasználóknak. Éppen ezért a tervezésre forditott idő és pénz megtérül, ha munkaigényesebb módszereket is kell használni.

I r o d a l o m

[1] P. H. Seaman, R. C. Soucy, Simulating operating systems. IBM Syst. J. 8, 4 (1969). 264-279.

[2] M.M. Mac Doughall, Computer system simulation: An introduction. Computer Survey 2, 3 (1970) 191-209.

[3] D. P. Gaver, Probability models for multiprogramming Computer Systems. Journal of ACM 14, 3 (1967) 423-438.

[4] F. Hanssmann, W. Kistler, H. Schulz, Modelling for computing center planning. IBM Syst. J. 10, 3 (1971) 305-524.

[5] B.R. Kirkerud, SIMBAS-A simple data base system. Norwegian Computing Center (1974) 19.

[6] M.I. Youchan, D.D. Rudie, E.J. Johson, The data processing system simulator (DPSS) Proc. AFIPS (1964) Fall Joint Computer Conf. 26, 251-276.

[7] G.S. Schedler, A queuing model of a multiprogrammed computer with a twolvele storage System. Comm. of ACM 16, 1 (1973) 3-10.

[8] W.I. Stanley, Measurement of system operational statistics. IBM Syst. J. 8,4 (1969) 299-308.

[9] L.R. Huesmann, R.P. Goldberg, Evaluating computer Systems throught simulation. Computer Journal 10, 2 (1967).

[10] N.R. Nielsen, The simulation of time-sharing systems. Comm. of ACM. 10, 7 (1967) 397-412.

[11] A. L. Scherr, An analysis of time- shared computer systems. MIT Press, Combridge, Mass. (1967).

[12] P.M. Seaman, On teleprocessing system desi design. Port VI. The role of the digital simulation. IBM Sys. J. 5,3 (1966) 175-189.

[13] Comutéer System Simualtion (CSS). H20-874-1 IBM DPD White Plarus, N.Y. (1972)

[14] E. Stauder, Program átfutási idők becslése szimulációs modellezéssel. Információ és Elektronika (1974) 9-14.

[15] ARTS-Analysis of Real–Time Systems. Cadis Project. University Stockholm (1972) 107.

[16] D.M. Braddock, C.B. Dowling, Simulation, Evaluation and Analysis Language (SEAL) IBM CPL.Prog. No. 360D 15.1.005

[17] G. K. Hutchinson, A computer center simulation project. Comm. ACM 8,9 (1975). 559-568.

[18] W. Chang, Single-server queueing processes in computing system. IBM. Syst. J. 9, 1 (1970) (1970) 36-71.

[19] Jo. Piene, Simulation of Computer Systems- Case Study. NCC pub. No. S 46. (1972) 113.

S u m m a r y

Planning and simulation modeling of computer system configurations

Ernő Stauder

The paper describes the simulation methods being able to evaluate and optimize the work of computer configurations.

Р Е З Ю М Е

ПЛАНИРОВАНИЕ И МОДЕЛИРОВАНИЕ
КОНФИГУРАЦИЙ СИСТЕМ ЭВМ

Е. Штаудер

В работе описываются методы моделирования, оценка и оптимизация работы конфигураций ЭВМ.