Ing.Jaroslav Plucar
VÚEPE Ostrava

PARAMETRICKÝ SYSTÉM PRO MANIPULACI S DATY PRO POTŘEBY JEDNORÁZOVÝCH AGREGACÍ INFORMACÍ Z EXISTUJÍCÍCH DATOVÝCH SOUBORŮ

1. Úvod.

Předmětem tohoto příspěvku je parametrický programový systém pro manipulaci s daty,který byl vytvořen z důvodu zefektivnění jednorázových aplikací matematických metod a jednorázových aplikací metod pro agregaci informací z agendových datových souborů,které jsou uloženy na paměťových médiích samočinných počítačů.

Systém byl vytvořen ve výzkumném ústavu ekonomiky paliv a energetiky (VÚEPE) Praha,pobočka Ostrava.Je implementován v jazyku PL/1 FULL a může pracovat prakticky na všech středních počítačích,které mají tento kompilátor, tzn. především na počítačích EC 1040 a IBM 360 a 370. Teprve v další fázi je plánováno jeho přeprogramování do Assembleru a jeho segmentace,aby mohl být využit i na menších počítačích, konkrétně na EC 1021.

Předkládaný příspěvek neobsahuje z důvodu omezeného rozsahu úplný popis systému.V kapitole 2 je uvedeno zdůvodnění proč byl vlastně systém vytvořen a ve zbývajících kapitolách jsou pak popsány základní principy systému a způsob, jakým jsou v rámci systému řešeny uživatelské úlohy.

2. Jednorázové agregace informací v podmínkách ASŘ.

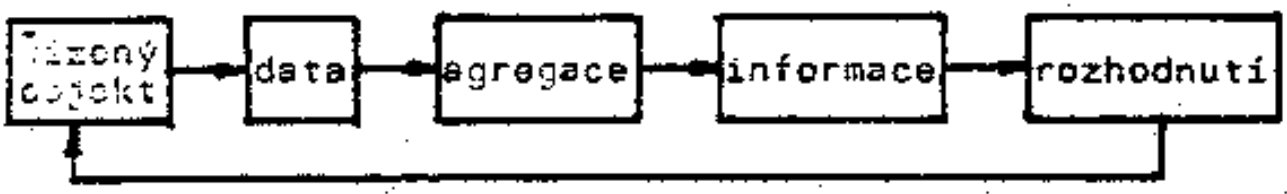
V současné době jsme svědky rozvoje metod,které by umožnily využít výpočetní techniky k vytváření podkladů pro potřeby rozhodování.Rozvoj těchto metod v současné fázi budování ASŘ není náhodný a souvisí se současnou "inovací" přístupů k vytváření a využívání ASŘ.

V posledních letech nastal totiž určitý odklon od původní (mechanistické) koncepce,která předpokládala poměrně široké automatizované použití rozhodovacích algoritmů na samočinných počítačích (názorně řečeno,počítač by měl rozhodovat sám bez zásahu řídícího pracovníka).V současné době se prosazuje koncepce,která ponechává rozhodování řídícím pracovníkům s tím,že počítač bude pouze pro řídící pracovníky připravovat agregované informace potřebné k rozhodnutí.

Tento odklon není náhodný.Souvisí s jakýmsi "myšlenkovým střetem" dvou škol (dvou strategií),které se zabývají metodami projektování a budování automatizovaných systémů (bližší viz /1/). Není samozřejmě účelem příspěvku,aby řešil tuto problematiku. Bylo ale nutno se o ní zmínit,protože zmíněný odklon bude mít za následek i dosti podstatnou změnu v názorech na využívání výpočetní techniky v procesu budování a využívání ASŘ.

2.1 Změna okruhu používaných metod.

Prvním důsledkem odklonu je změna okruhu metod,které by měly být využívány při budování a využívání ASŘ.Zjednodušené blokové schema rozhodovacího procesu je uvedeno na obr. 1 na následující straně.Původní (mechanistická) koncepce kladla stejný důraz na automatizaci bloků "agregace" i "rozhodnutí". Nyní je kladen důraz pouze na automatizaci bloku "agregace", z něhož by měly vycházet (formou tiskových sestav nebo na displeji) podkladové informace pro rozhodování.



obr. 1

Paleta existujících metod pro agregaci informací je velmi široká, od nejjednodušších, až po ty nejsložitější. Např. pouhé setřídění údajů a jejich vytištění do tabulky už představuje určitou agregaci. Hlavní úloha v agregačních metodách ale bezesporu patří matematickým metodám. Při chápání úlohy matematických metod se ale většinou dopouštíme jedné zásadní chyby: pod pojmem matematické metody si představujeme jen ty složitější, např. síťovou analýzu, faktorovou analýzu, metodu GUHA (metoda pro generování hypotéz) apod. Výsledky dané těmito metodami pak považujeme za definitivní, řídícím pracovníkem dále nezměnitelné. Takové chápání je chyb[illegible] ze dvou důvodů:

- Paleta existujících matematických metod je podstatně širší od nejjednodušších, běžně používaných (např. výpočet maxima, minima, průměru, sumace) až po ty nejsložitější, jejichž příklady už byly uvedeny.
- Výsledky matematických metod, i těch nejsložitějších, představují pouze podklad pro rozhodnutí a nemohou být akceptovány dříve, dokud je neschválí příslušný řídící pracovník. Jinými slovy, je nutno zdůraznit poznávací funkci matematických metod na úkor jejich rozhodovací funkce. Ocitujme /7/: "Jak málo se může s matematickými metodami nalézt hotové rozhodnutí, tak velký může být na druhé straně příspěvek matematických metod k procesu poznání, který předchází rozhodnutí".

Problém volby vhodných agregačních metod (ani problému odklonu od rozhodovacích algoritmů) není rozhodně problémem programátorským a zmiňuji se o něm jen z toho důvodu, že v současných programech jsou používány jen nejjednodušší agregační metody, jejichž použití je navíc do značné míry zatíženo praktikismem. Při vhodném a odborně fundovaném použití agregačních metod bychom byli schopni z údajů, uložených v současné době na paměťových médiích počítače, získat mnohem větší množství informací potřebných pro rozhodování a řízení než dosud.

2.2 Změna formy využívání agregačních metod.

Druhým důsledkem citovaného odklonu je předpokládaná změna formy využívání agregačních metod. V původní (mechanistické) koncepci byl kladen hlavní důraz na rutinní, periodicky se opakující aplikace. Nyní je vedle rutinních aplikací kladen důraz i na jednorázové aplikace, t.j. takové aplikace, které se vůbec neopakují nebo jejichž opakování je minimální. Při jednorázových aplikacích vystupují do popředí sémanticky prázdné (v současné době především parametrické) systémy, které umožňují jednoduchý a rychlý zápis zadané úlohy do parametrů. Při jednorázových aplikacích nehledíme tolik na efektivnost programů, jako tomu bylo u rutinních aplikací (efektivnost parametrických programů je samozřejmě nižší a spotřeba strojového času vyšší než u specializovaných, rutinních, programů).

Při řešení problematiky jednorázových agregací informací se až na malé výjimky nesetkáme s problémy v oblasti metod (z hlediska programů, které tyto metody aplikují je jedno, jsou-li zpracovávány jednorázově nebo periodicky). Setkáme se zde ale s dosti závažným problémem dosažitelnosti údajů. Většina existujících programů, které aplikují některou agregační metodu předpokládá, že všechny potřebné údaje budou vstupovat v jednom datovém souboru.

V praxi se ale většinou setkáváme s případy, kdy údaje potřebné pro agregaci jsou uloženy ve dvou,nebo více agendových datových souborech. Musíme tedy nejdříve "přemístit" údaje ze dvou nebo více datových souborů do jednoho nového souboru, který pak slouží jako vstupní pro příslušný aplikační program. V současné době je "přemísťování" většinou řešeno vytvořením a odladěním speciálních programů. Takové řešení klade ovšem značné nároky na programátorskou kapacitu a zároveň podstatně prodlužuje realizační dobu úlohy na počítači (t.j. čas, který uplyne od zadání požadavku do obdržení výsledků z počítače).

Problém dosažitelnosti údajů by měl být automaticky vyřešen výstavbou datových bank. V současné době je ovšem převážná většina datových bank teprve ve stádiu projektu a jeví se proto účelným vytvoření takového parametrického systému,který by na základě jednoduchých parametrů dokázal "přemístit" údaje ze dvou nebo více existujících datových souborů do jednoho nového souboru a nahradil tím programátorskou kapacitu potřebnou k vytváření "přemísťovacích" programů.

Jak ukazují dosavadní zkušenosti, požadavky na jednorázové agregace informací jsou ze strany řídících pracovníků předkládány,ale až na nepatrné výjimky nejsou realizovány ať už z důvodu nedostatečné programátorské kapacity nebo z důvodu dlouhé realizační doby (názorně řečeno,řídící pracovník by dostal výsledky z počítače až v době,kdy už by je nepotřeboval).

3. Princip systému.

Jak vyplývá z kapitoly 2, je hlavním úkolem systému pro manipulaci s daty podstatným způsobem snížit pracnost spojenou s "přemisťováním" údajů ze dvou nebo více datových souborů do jednoho souboru a tím také zkrátit realizační dobu úlohy na počítači.

Tohoto efektu bylo dosaženo volbou způsobu zadávání parametrů, při němž je možno pracovat se jménem souboru jako s argumentem. Názorně řečeno, máme-li potřebné údaje uloženy ve dvou datových souborech, pak pro jejich "přemístění" do nového souboru postačí jediný parametrový štítek, na kterém jsou uvedena jména všech tří souborů, t.j. nového a obou původních. Jediný parametrový štítek tedy nahradí speciální program, který by pro tento účel bylo nutno vytvořit. Podrobnosti o zadávání parametrů jsou uvedeny v odst. 4.2.

Na základě takto zadaného parametrového štítku je pak "vygenerován" program pro přemístění potřebných údajů do nového souboru (nejde samozřejmě o generování v pravém slova smyslu, příslušné programy jsou parametrické). "Vygenerovaný" program pracuje na principu normalizovaného programování. Informace pro vygenerování klíčů, porovnávacích výroků a výroků pro přemístění hodnot jednotlivých údajů jsou získány z popisů všech tří souborů, t.j. nového a obou původních. Podrobnosti o popisování datového souboru jsou uvedeny v odst. 4.1.

Kromě uspořené programátorské kapacity získáno tímto přístupem především podstatné zkrácení realizační doby úlohy na počítači (u běžných úloh na dva až tři dny), takže agregované informace jsou k dispozici v době, kdy řešený problém je ještě "živý". Nepřímým důsledkem tohoto způsobu zadávání parametrů je i podstatné snížení rizika výskytu chyby (v deseti až patnácti parametrových štítcích, které reprezentují zadání jedné úlohy, se lze stěží dopustit chyby).

4. Popis systému.

Systém je tvořen pěti programy. Nosným programem systému je unifikovaný aktualizační program, který provádí "přemísťování" údajů ze dvou nebo více existujících datových souborů do nového souboru. Dalším důležitým programem je unifikovaný transformační program, který transformuje běžné agendové soubory (a soubory vytvořené unifikovaným aktualizačním programem) do formy, na kterou jsou "zvyklé" programy pro aplikaci agregačních (většinou matematických) metod. Další důležitá úloha transformačního programu spočívá v tom, že umožňuje "minimalizovat" dodědrovávaná data. Ne všechny údaje, které jsou třeba pro realizaci agregační úlohy, jsou totiž sledovány na počítači. Takové údaje je třeba dodědrovat a z pochopitelných důvodů má uživatel zájem na co nejkratším zápisu takových údajů (čím menší je objem dat, tím menší je pracnost spojená s jejich zápisem do formulářů a tím menší je také riziko výskytu chyby, ať už při zápisu nebo při dědrování). Pro zkrácení zápisu je možno např. ukládat do věty intervaly hodnot, při doplňování nového údaje do všech vět souboru není nutno zapisovat identifikační údaje apod.

Zbývající programy jsou pouze kontrolní a obslužné. Vzhledem k omezenému rozsahu příspěvku nejsou proto popsány, stejně jako není popsán transformační program. Zbývající část této kapitoly je věnována popisu funkce unifikovaného aktualizačního programu, který je nejdůležitějším programem celého systému pro manipulaci s daty.

4.1 Popis datového souboru.

Základním prvkem, na němž je založena funkce aktualizačního programu (i všech ostatních programů systému) je popis datového souboru. V popisu jsou uloženy všechny důležité informace o datovém souboru (jeho jméno, délka věty, seznam údajů, jejich uložení ve větě apod. - viz dále). Díky popisu datového souboru nůžeme při zadávání parametrů

pracovat se jménem souboru jako s argumentem,což podstatným způsobem zmenšuje objem zadávaných parametrů (jak už bylo uvedeno v kapitole 3). Pro úplnost je třeba uvést, že stávající verze systému pracuje pouze se sekvenčními soubory, kde údaje jsou ve větách uloženy poziční.

Popis datového souboru je reprezentován děrnoštítkovým souborem, který se skládá z jednoho úvodního a n-údajových štítků.Na úvodním štítku je uvedeno jméno souboru,délka věty a některé další charakteristiky souboru.Za úvodním štítkem následují údajové štítky.Informace o jednom údaji uloženém ve větě datového souboru jsou uvedeny na jednom údajovém štítku. Jedná se o následující informace:

- Symbolické jméno údaje.
- Informace o fyzickém uložení údaje ve větě,t.j. pozice údaje,jeho délka v bytech,typ fyzického uložení (znakové,binární apod.),dimenze pole (je-li ve větě pro jeden údaj uloženo více hodnot,např."jazykové znalosti" ve větě souboru JEP).
- Logické charakteristiky údaje - jedná se o neuzavřenou množinu informací,které mají sémantický charakter. Ve stávající verzi systému je zatím používána pouze tzv. "metodická charakteristika", která udává (ve smyslu matematické statistiky) zda jde o údaj nominální, ordinální nebo kardinální. Tato charakteristika rovněž určuje,jde-li o údaj kvalitativní (kódový) nebo kvantitativní (početní).
- Určení stromové struktury údajů v datovém souboru,t.j. určení hierarchické úrovně identifikačního údaje a typu setřídění (vzestupně nebo sestupně) - viz /5/.
- Plné jméno údaje (pro potřeby uživatele).

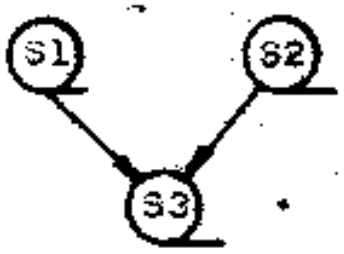
V praxi se dosti často stává,že v jednom datovém souboru existuje více typů vět různé délky a s různým rozmístěním údajů. Pro takový datový soubor je třeba vytvořit tolik popisů,kolik typů vět je v datovém souboru obsaženo.

Zavádíme proto pojmy fyzický soubor a aplikační soubor. Jeden aplikační soubor je tvořen všemi větami stejného typu, uloženými ve fyzickém souboru. Popis datového souboru je tedy vztažen na aplikační, nikoliv na fyzický soubor.

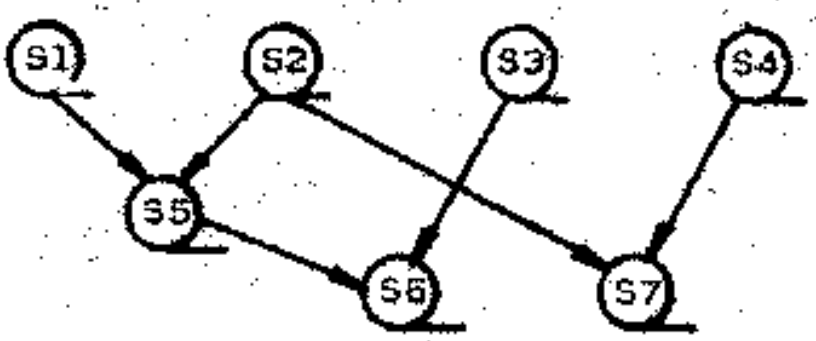
Pojem aplikační soubor se používá i v oblasti datových bank (viz /4/ a /5/, český přehled v /3/). Je ovšem nutno podotknout, že v případě popisovaného systému se nejedná o práci s bankou dat, ale pouze o využití databankových přístupů při volbě vhodných metod pro práci se sekvenčními agendovými soubory. Databankové prvky jsou využity i v jiných částech systému. Jak vidíme na první pohled, plní popis datového souboru v rámci popisovaného systému obdobnou funkci jako popis aplikačního souboru jazykem DDL (Data Description Language) v rámci datových bank.

4.2 Dekompozice úlohy a zadávání parametrů.

Nejme úlohu, při které máme do N-výstupních souborů přenístit údaje z M-vstupních souborů. Takovou úlohu musíme dekomponovat na elementární aktualizační úlohy, kde jedna elementární aktualizační úloha přenístí do jednoho výstupního souboru údaje ze dvou vstupních souborů. Každá úloha se tedy "rozpadne" na P-elementárních úloh. Každé elementární aktualizační úloze odpovídá jeden zadaný parametrový štítek. Schema elementární úlohy je uvedeno na obr. 2, příklad dekompozice úlohy (vymyšlené) na obr. 3.



obr. 2



obr. 3

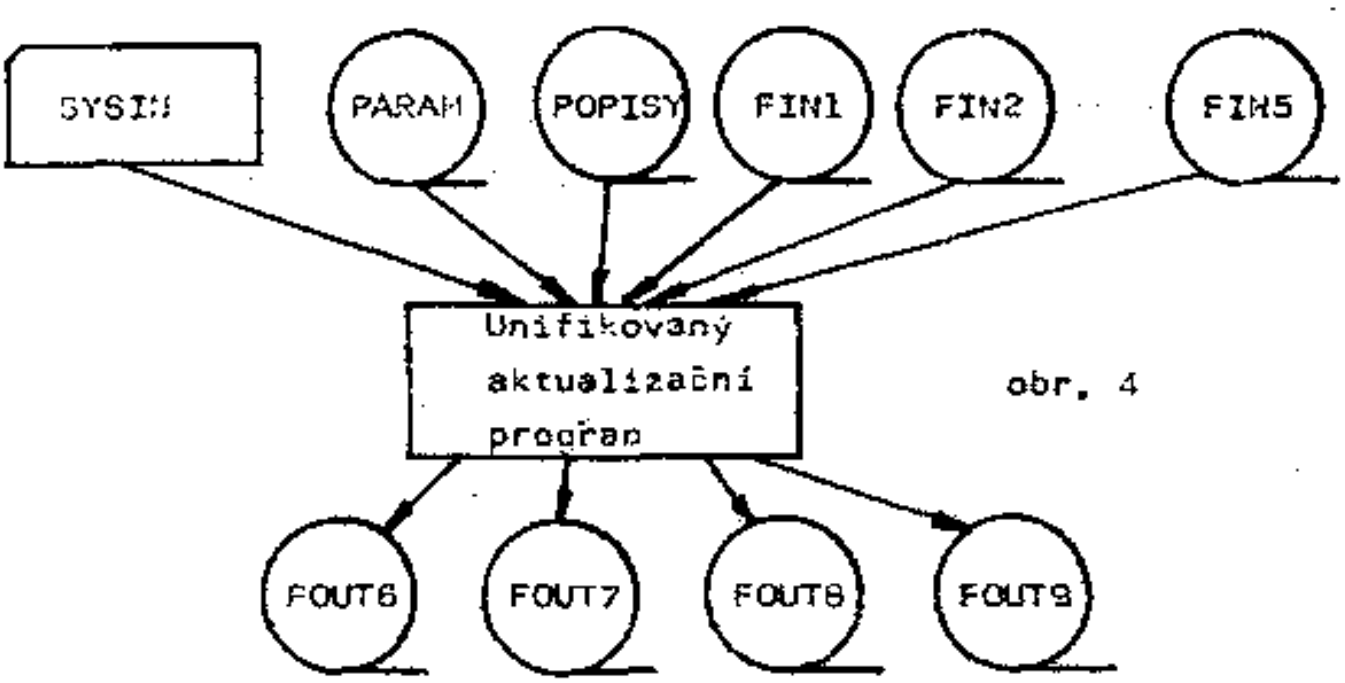
Vymyšlená úloha z obr. 3 (čtyři vstupní soubory,dva výstupní) je dekomponována na tři elementární úlohy (pro její zadání budou tedy třeba tři parametrové štítky):

1. Soubor S5 vytvořit ze souborů S1 a S2.
2. Soubor S6 vytvořit ze souborů S5 a S3.
3. Soubor S7 vytvořit ze souborů S2 a S4.

Soubor S5 je pouze pracovní,slouží jen k vytvoření souboru S6. Takové pracovní soubory nemusíme vypisovat na výstupní médium (do "čísla nosiče" takového souboru na parametrovém štítku zapisujeme nulu - viz odst. 4.3).

Každý zkušenější programátor ví,jak složité mohou být vztahy mezi dvěma soubory při jejich interakci. Popisovaný systém nemůže samozřejmě předpokládat všechny možnosti, akceptuje jen ty, se kterými se setkáváme v běžné praxi. Jedním z nejdůležitějších je vztah klíčů obou vstupních souborů. Systém akceptuje jednak totožné klíče,dále klíče s různou délkou (např. chceme-li do "závodových" vět doplnit sumarizované údaje z vět "podnikových") a konečně klíče s odlišnou strukturou (např. pro doplňování údajů do datové věty z číselníku). S různými typy interakce mezi dvěma vstupními soubory se systém vyrovnává pomocí tzv. "typu aktualizace", jehož hodnota je zadávána v parametrovém štítku pro každou elementární úlohu zvlášť.Při té příležitosti je třeba uvést, že pojem aktualizace je v rámci popisovaného systému chápán poněkud obecněji než v běžné praxi. Pod pojmem aktualizace zde rozumíme modifikaci úrovně informace uložené v původním datovém souboru na základě informací uložených ve změnovém souboru. V současné verzi systému je možno zadávat tři typy aktualizace, a to novelizaci informací (kryje se s běžným chápáním pojmu aktualizace), dále slučování informací (jakýsi "logický merging", kdy merging se nezastavuje na úrovni vět,ale pokračuje na úroveň údajů a polí údajů, např. jsou sečteny hodnoty kvantitativních údajů apod.), a konečně vylučování informací, které je v podstatě reciproční operací ke slučování.

4.3 Stručný popis činnosti aktualizačního programu.



obr. 4

Program nejdříve načítá parametrové štítky ze souboru SYSIN. Parametry je možno uložit i do kmenového souboru PARAM a z něho je pak vyvolat (v tom případě je v souboru SYSIN přítomen jediný štítek se jménem parametrů). Po načtení a kontrole zadaných parametrů načte program popisy všech souborů,které figurují ve zpracovávané úloze, ze souboru POPISY. Po načtení popisů jsou generovány řídicí tabulky pro obsazení klíčů,testy klíčů a přesuny údajů. Po vygenerování tabulek je pak zpracována úloha.Pro vstupní soubory je rezervováno pět DD-jmen (FIN1-FIN5), pro výstupní čtyři (FOUT6-FOUT9).

Umístění souborů na jednotlivá DD-jména je řízeno z parametrových štítků,kde u každého jména souboru je kolonka "číslo nosiče". Uvedeme-li do ní číslo "2",znamená to, že soubor je vstupní a bude "nasazen" na DD-jméno FIN2,Uvedeme-li do ní číslo "7",znamená to, že soubor je výstupní a bude vypsán na DD-jméno FOUT7.

Jak je jistě z popisu (byť velmi stručného) patrno,nemůže systém obsluhovat začátečník. Pro obsluhu jsou potřebné určité zkušenosti v práci s datovými soubory a s normalizovaným programováním.

5. Závěr.

I když systém vypadá na první pohled poměrně atraktivně (nahrazení poměrně komplikovaných programů pěti až deseti parametrovými štítky), je třeba se vyvarovat přehnaného optimismu. Systém rozhodně není určen pro periodické(rutinní) aplikace a může tedy nahradit dosavadní "aktualizační" programy jen do určité míry. Důvodem jsou jednak implementační omezení běžná u parametrických programů a dále vyšší spotřeba strojového času ve srovnání se specializovanými programy.

Systém je určen pro jednorázové aplikace a tím je také poměrně jednoznačně vymezen okruh jeho možného použití.

Literatura:

/1/ Pivoda,M.,Plucar,J.,Vokurka,J.:
Zadávání parametrizovaných úloh při práci s bankou dat.
MAA 2. s 3./1978.

/2/ Martha Hyval Jones:
HIPO FOR DEVELOPING SPECIFICATION.
Datamation 3./1976.

/3/ Brabenec,I.,Žďárský,J.:
K vývoji databázových systémů.
MAA 5./1975.

/4/ Engles,R.W.:
A TUTORIAL ON DATA-BASE ORGANISATION.
Annual review in automatic programming,vol.7,part 1,1972.

/5/ Buchholz,W.:
FILE ORGANISATION AND ADRESSING.
IBM systems journal,vol. 2,1963.

/6/ Černý,V.:
Klasifikace a identifikace strojového zpracování informací.
MAA 9./1977.

/7/ Rheinboldt,W.C.:
Matematische methoden und Entscheidungen in Leitun soerate.
Die Wiertecheft, 1./1978.