MEDYCYNA

BIULETYN TECHNICZNO-INFORMACYJNY

6(252)
1983

Warunki prenumeraty

Jednostki gospodarki uspołecznionej, instytucje, organizacje i wszelkiego rodzaju zakłady pracy zamawiają prenumeratę w miejscowych Oddziałach RSW "Prasa-Książka-Ruch", w miejscowościach zaś, w których nie ma Oddziałów RSW – w urzędach pocztowych. Czytelnicy indywidualni opłacają prenumeratę wyłącznie w urzędach pocztowych i u doręczycieli. Prenumeratę roczną w cenie 1896 zł należy zamawiać do 25 listopada na rok następny, półroczną do 10 czerwca na II półrocze.

Cena 158 zł

ZRZESZENIE PRODUCENTÓW ŚRODKÓW INFORMATYKI, AUTOMATYKI i APARATURY POMIAROWEJ „MERA"

BIULETYN TECHNICZNO-INFORMACYJNY

Warszawa, czerwiec 1983

S P I S T R E Ś C I

Spis artykułów "Pomiary-Automatyka-Kontrola" nr 4-5/1983

Opracowanie: Redakcja Biuletynu Techniczno-Informacyjnego "Mera", ul. Poezji 19, 04-994 Warszawa /tel. 12-90-11 wew. 17-54/. Wydawca: Przedsiębiorstwo Automatyki Przemysłowej "Mera-Pnefal", ul. Poezji 19, 04-994 Warszawa. Zam. 145/83. Nakład 1150 egz.

dr JANUSZ WOJDYŁA

PRZEGLĄD I CHARAKTERYSTYKA METOD KOMPRESJI BAZ DANYCH

Wyróżnianie i opis różnych obiektów informatycznych w systemach baz danych jest zagadnieniem złożonym ze względu na małą precyzję języka naturalnego. Zachodzi zatem konieczność kodowania danych czyli przyporządkowania słów utworzonych z elementów skończonego alfabetu semantycznym obiektom kodowanej zbiorowości w relacji jeden do jednego. Czynnością odwrotną procesu kodowania jest dekodowanie.

W procesach przetwarzania danych kodowanie i towarzyszące mu dekodowanie jest wykonywane wielokrotnie, często stanowiąc swoisty łańcuch procesów kodowania. Kodowanie pierwotne zachodzi wówczas, gdy nadajemy /zgodnie z przyjętym systemem kodowania/ symbole wyróżnionym elementom kodowanej zbiorowości. Prezentacja danych na nośnikach technicznych wymaga najczęściej kolejnego, wtórnego procesu kodowania wartości danych /symboli/ w ciągi elementarnych sygnałów binarnych na nośnikach. O ile kodowanie pierwotne jest wykonywane przez człowieka, to kodowanie wtórne wykonywane jest automatycznie przez urządzenia informatyczne.

W procesie kodowania wtórnego wybór systemu kodowania jest znacznie ograniczony przez producenta urządzenia kodującego. Najczęściej stosowanymi obecnie systemami kodowania są EBCDIC oraz ANSCII [1/]

Jeśli znana jest liczebność elementów kodowanej zbiorowości /q/ oraz liczba możliwych symboli /L/:

$$L = D^{li}$$

gdzie:
D - liczba znaków alfabetu,
li - długość symbolu,
to, przy założeniu stałej długości symboli, możemy obliczyć sprawność kodowania /S_k/.

$$S_k = \frac{q}{L} \times 100\%$$

Używając dwuznakowego kodu numerycznego do przedstawienia 25 różnych wartości semantycznych uzyskujemy sprawność kodowania rzędu 25%. Różnica między liczbą możliwych do wystąpienia symboli /czyli tzw. siłą leksykograficzną języka kodowania/ a liczbą kodowanych elementów /L-q/ stanowi nadmiar bezwzględny czyli redundancję /R_b/

$$R_b = L-q$$

W omawianym przykładzie redundantnych będzie 75 elementów utworzonego kodu [2/].

Przechowywanie baz danych w pamięciach zewnętrznych komputerów jest zawsze obciążone redundancją, która wynika z kilku przyczyn:

1. Przyjętego kodu binarnej reprezentacji danych na magnetycznych nośnikach pamięci. Przykładowo z 256 możliwych kombinacji kodu EBCDIC w praktyce przechowywania danych gospodarczych najczęściej nie używa się więcej niż 64 kombinacji odpowiadających znakom cyfr, dużych liter oraz wybranych znaków specjalnych, co ilustruje tablica 1.
2. Umieszczenia zmiennych wartości danych w polu o stałej długości, równej długości wartości najdłuższej.
3. Tworzenia statycznych pól wielokrotnych.
4. Występowania jednej lub kilku wartości danych o wyjątkowo dużej częstości.
5. Występowania istotnej korelacji między kolejnymi wartościami danych.

Pomijając przypadki celowego wprowadzania redundancji do baz danych /np. różne cyfry, bity i liczby kontrolne /51 s.160/, gene-

---

1/ EBCDIC - ang. Extended Binary-Coded Decimal Interchange Code.

ANSCII - American National Standard Code for Information Interchange /dawniej USACII/.

Obszerny wykaz stosowanych wtórnych systemów kodowania znajduje się w [49].

2/ Czytelników zainteresowanych poszerzeniem lub przypomnieniem sobie wiadomości z zakresu teorii informacji i kodowania odsyła się do prac [2] lub [60].

Tabela 1

Wyniki analizy częstości występowania znaków w przykładowym zbirze danych gospodarczych / z zakresu planowania produkcji /

KOD	ZNAK	CZĘSTOŚĆ	KOD	ZNAK	CZĘSTOŚĆ	KOD	ZNAK	CZĘSTOŚĆ	KOD	ZNAK	CZĘSTOŚĆ
40	SP	,50782	60	`	,00000+	80		,00000+	C0		,00000+
30	0	,12552	61	a	,00000+	81		,00000+	04	PE	,00000+
31	1	,04864	62	b	,00000+	82		,00000+	05	HT	,00000+
32	2	,03479	63	c	,00000+	83		,00000+	06	LF	,00000+
34	4	,01913	64	d	,00000+	84		,00000+	07	DEL	,00000+
33	3	,01715	65	e	,00000+	85		,00000+	08		,00000+
35	5	,01543	66	f	,00000+	86		,00000+	09		,00000+
36	6	,01535	67	g	,00000+	87		,00000+	0A	SMM	,00000+
39	9	,01213	68	h	,00000+	88		,00000+	0B	VT	,00000+
38	8	,01210	69	i	,00000+	89		,00000+	0C	FF	,00000+
37	7	,01044	6A	[	,00000+	8A		,00000+	CA		,00000+
42	K	,01044	6B	,CM	,00000+	8B		,00000+	CB		,00000+
3C	P	,01001	6C	<	,00000+	8C		,00000+	CC		,00000+
39	9	,00824	6D	(	,00000+	8D		,00000+	CD		,00000+
2E	S	,00758	6E	+	,00000+	8E		,00000+	CE		,00000+
3A	Z	,00705	6F	,CR	,00000+	8F		,00000+	CF		,00000+
41	A	,00565	70		,00000+	90		,00000+	D0		,00000+
25	E	,00561	71	q	,00000+	91		,00000+	0D	CR	,00000+
35	W	,00523	72	r	,00000+	92		,00000+	0E	SO	,00000+
43	C	,00516	73	s	,00000+	93		,00000+	0F	SI	,00000+
4E	.	,00405	74	t	,00000+	94		,00000+	10	DLE	,00000+
31	A	,00404	75	u	,00000+	95		,00000+	11	DC1	,00000+
37	G	,00399	76	v	,00000+	96		,00000+	12	DC2	,00000+
3B	O	,00366	77	w	,00000+	97		,00000+	13	TM	,00000+
45	X	,00290	78	x	,00000+	98		,00000+	D8		,00000+
3W	R	,00279	79	y	,00000+	99		,00000+	14	RES	,00000+
44	D	,00252	7A	z	,00000+	9A		,00000+	DA		,00000+
53	T	,00240	7B	,CL	,00000+	9B		,00000+	DB		,00000+
4B	L	,00224	7C	*	,00000+	9C		,00000+	DC		,00000+
46	H	,00223	7D	)	,00000+	9D		,00000+	DD		,00000+
42	B	,00149	7E	;	,00000+	9E		,00000+	DE		,00000+
48	N	,00132	7F	~	,00000+	9F		,00000+	DF		,00000+
45	Y	,00115	02	STX	,00000+	A0		,00000+	E0		,00000+
4F	U	,00058	03	ETX	,00000+	A1		,00000+	E1		,00000+
46	F	,00046	62		,00000+	A2		,00000+	15	NL	,00000+
41	J	,00046	63		,00000+	A3		,00000+	16	BS	,00000+
5D	-	,00016	64		,00000+	A4		,00000+	17	IL	,00000+
61	/	,00001	65		,00000+	A5		,00000+	E5		,00000+
20	EIB	,00000+	66		,00000+	A6		,00000+	18	CAN	,00000+
27	ESC	,00000+	67		,00000+	A7		,00000+	19	EM	,00000+
2F		,00000+	68		,00000+	A8		,00000+	1A	CC	,00000+
29		,00000+	69		,00000+	A9		,00000+	1B	CU1	,00000+
24	SH	,00000+	6A		,00000+	AA		,00000+	EA		,00000+
2F	CU2	,00000+	6B	,	,00000+	AB		,00000+	FB		,00000+
26		,00000+	6C	%	,00000+	AC		,00000+	EC		,00000+
2D	ENQ	,00000+	6D	,DE	,00000+	AD		,00000+	FD		,00000+
2E	ACK	,00000+	6E	>	,00000+	AE		,00000+	EE		,00000+
2F	BEL	,00000+	6F	?	,00000+	AF		,00000+	FF		,00000+
30		,00000+	70		,00000+	B0		,00000+	1C	IFS	,00000+
31		,00000+	71		,00000+	B1		,00000+	1D	IGS	,00000+
32	SYN	,00000+	72		,00000+	B2		,00000+	1E	IRS	,00000+
33		,00000+	73		,00000+	B3		,00000+	1F	IUS	,00000+
34	PN	,00000+	74		,00000+	B4		,00000+	20	DS	,00000+
35	RS	,00000+	75		,00000+	B5		,00000+	21	SOS	,00000+
36	UC	,00000+	76		,00000+	B6		,00000+	22	FS	,00000+
37	EOT	,00000+	77		,00000+	B7		,00000+	23		,00000+
38		,00000+	78		,00000+	B8		,00000+	24	BYP	,00000+
39		,00000+	79		,00000+	B9		,00000+	25	LF	,00000+
3A		,00000+	7A	:	,00000+	BA		,00000+	FA		,00000+
3B	CU3	,00000+	7B	#	,00000+	BB		,00000+	FB		,00000+
3C	DC4	,00000+	7C	@	,00000+	BC		,00000+	FC		,00000+
3D	NAK	,00000+	7D	'	,00000+	BD		,00000+	FD		,00000+
3E		,00000+	7E	=	,00000+	BE		,00000+	FE		,00000+
3F	SUB	,00000+	7F	"	,00000+	BF		,00000+	FF		,00000+

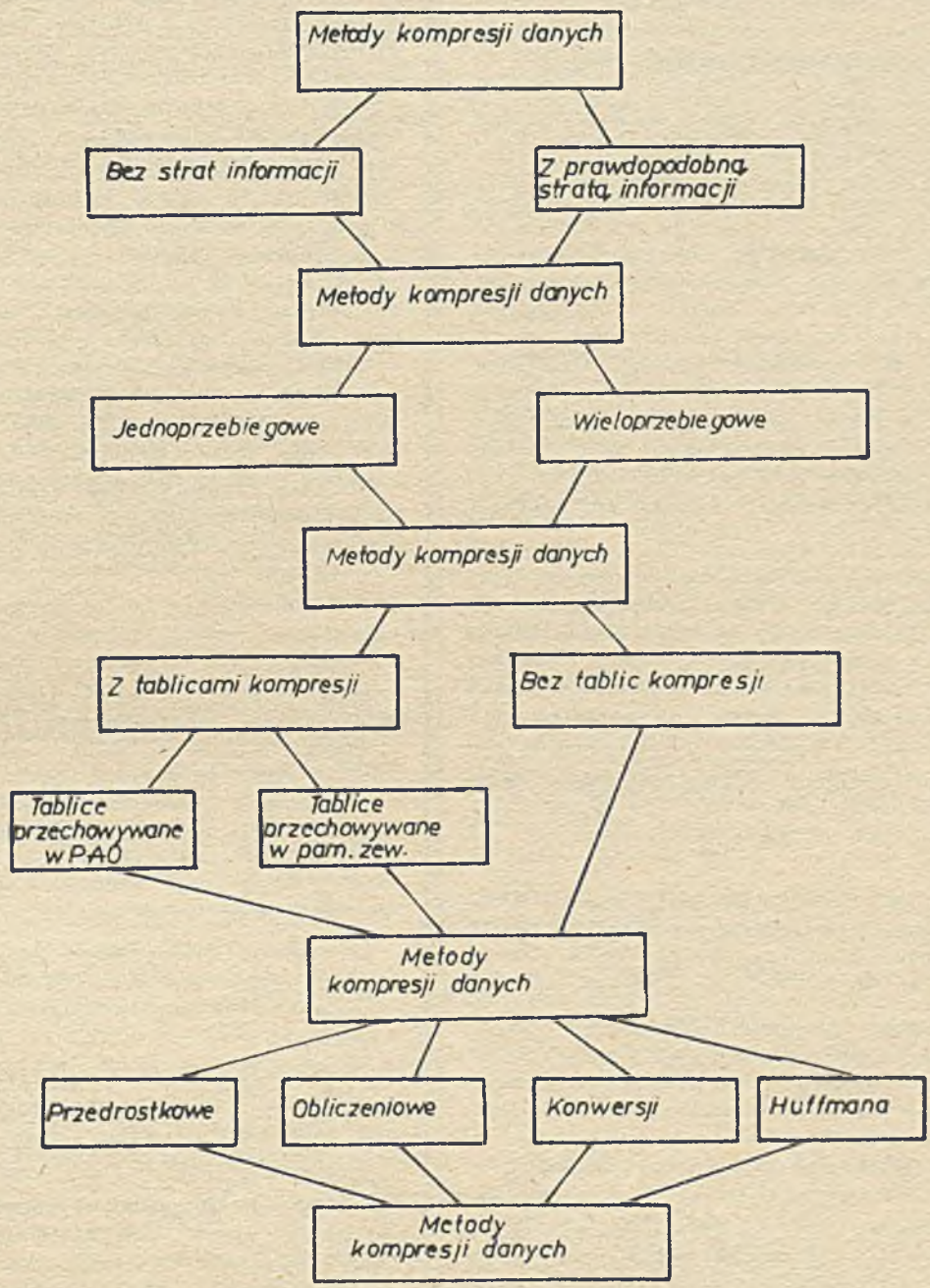
KP-KRESKA POZIOMA,WK-WYKRZYKNIK,DL-ZNAK DOLARA,PK-PODKRESLENIE
SP-SPACJA,ZN-ZNAK CENTA,+ -BRAK KODU NIE WYSTAPIL ANI RAZU

ralnie jest ona w systemach informatycznych zbyteczna i należy ją ograniczać.

Zakładając, że drogą zmniejszenia redundancji jest skrócenie długości symboli lub ograniczenie przyjętego zbioru znaków kodu należy wprowadzić dodatkowe pojęcie zagęszczania danych /ang. data compaction/. Zagęszczenie danych jest formą wtórnego kodowania danych, która zmniejsza obszar pamięci zajmowany przez dane chroniąc jednocześnie informacje uważane za istotne[3]. Zagęszczanie danych nie zawsze musi być procesem odwracalnym, czego przykładem może być algorytm kompresji znaków klucza zaproponowany przez H.K. Changa /11/, a zastosowane w organizacji skorowidzów metody dostępu VSAM /73/ i /67/.

Pewną formą zagęszczania danych jest abrewiacja polegająca na skracaniu nazw wyrażonych w języku naturalnym dzięki opuszczaniu niektórych fraz lub liter[4]. Efekt

3/ Przykładami kodów zagęszczających wartości identyfikatorów są kody fonetyczne /np. SOUNDEX, SABRE i ARF/ opisane przez J.L Dolby'ego /15/ oraz G.Wiederholda /72/.



Rys. 1. Graf klasyfikacji metod kompresji danych ze względu na algorytm funkcji kompresji

zagęszczenia danych powstaje również w wyniku stosowania niektórych metod szyfrowania czyli odwracalnego kodowania danych w celu ukrycia treści informacji[5].

Reasumując, możemy przyjąć definicję procesu kompresji danych o następującej postaci. K o m p r e s j a d a n y c h jest odwracalnym procesem wtórnego kodowania danych realizowanym w celu zmniejszenia ich redundancji. Innymi słowy jest to proces odwracalnego zagęszczania danych. Proces przywracający danym ich pierwotną postać /z postaci skompresowanej/ określamy mianem dekompresji danych.

Klasyfikacja metod kompresji danych

Jeżeli przyjmiemy, że kompresja danych jest funkcją na elementach bazy danych to możemy ją zapisać symbolicznie jako: y = f/x/, gdzie x jest określonym elementem bazy danych. Wybrany element może wchodzić w skład sktruktury logicznej bazy /np. pole, rekord/ albo struktury fizycznej /np. słowo maszynowe, bajt, n-tka bajtów, bit lub n-tka bitów/. Zadaniem funkcji $\forall\forall f\forall\forall$ jest zatem odwzorowanie elementu $\forall\forall x\forall\forall$ w inny element $\forall\forall y\forall\forall$. Dziedziną funkcji jest zbiór elementów x, na których ona działa a zakresem zbiór elementów y, stanowiących wynik działania funkcji. Stąd każda metoda kompresji może być scharakteryzowana przez właściwą jej funkcji dziedzinę, zakres i algorytm.

Pierwszym kryterium klasyfikacji jest typ dziedziny, której elementy mogą być traktowane jako zbiór wartości semantycznych. Metody kompresji będziemy określać mianem semantycznych, jeśli będą one działały na zbiorze wartości elementów danych, uwzględniając znaczenie tych wartości w rozpatrywanym procesie przetwarzania danych. Pozostałe metody kompresji danych będziemy określać mianem syntaktycznych lub niesemantycznych.

Kolejnym kryterium klasyfikacji metod kompresji danych jest rodzaj algorytmu realizującego funkcję kompresji. Pełny wielokryteriowy graf klasyfikacji znajduje się na rys.1.

Pierwszym kryterium klasyfikacji stopnia jest dokładność procesu kompresji. Jeśli wielokrotna kompresja i dekompresja bazy danych umożliwia uzyskanie dokładnego pierwowzoru, to mamy do czynienia z algorytmem nie powodującym strat informacji /ang. information-lossless/[6].

Ze względu na liczbę przebiegów czytania zbioru kompresowanego algorytmy dzielimy na jedno i wieloprzebiegowe. Algorytmy jednoprzebiegowe umożliwiają wykonanie procesu kompresji w jednym przebiegu czytania kompresowanego zbioru danych. Do algorytmów jednoprzebiegowych zalicza się również te algorytmy, które wymagają wstępnych badań statystycznych zbioru danych. Algorytmy wieloprzebiegowe umożliwiają często osiągnięcie lepszych współczynników kompresji ale z uwagi na znaczny czas tego procesu nie mogą być brane pod uwagę przy kompresji baz danych o charakterze operatywnym[7].

Kolejnym kryterium jest konieczność używania przez algorytm różnego rodzaju tablic pomocniczych zwanych dalej tablicami kompresji. Najczęściej są to tablice konwersji znaków, fraz lub słów ale mogą również zawierać opisy kompresowanych zbiorów, a w tym wykazy pól niekompresowanych, nazwę algorytmu kompresji, itp. Przykładem algorytmu nie wymagającego tablicy jest algorytm usuwania nieznaczących zer i spacji lub algorytm korzystający ze specyfiki konstrukcji kodu EBCDIC /46/. Tablice kompresji mogą być przechowywane w specjalnym zbiorze, mogą występować jako pierwszy rekord każdego zbioru danych lub znajdować się w programie realizującym dany algorytm. W trakcie kompresji baz danych tablica winna znajdować się w pamięci operacyjnej. Algorytmy wymieniające fragmenty tablic kompresji w trakcie działania nie powinny być brane pod uwagę w projektowaniu kompresji baz danych.

Ostatnim z istotnych kryteriów podziału jest typ stosowanego algorytmu kompresji. Wyróżnia się cztery zasadnicze typy algorytmów:
- przedrostkowe,
- obliczeniowe,
- konwersji,
- Huffmana.

Algorytmy przedrostkowe umieszczają na początku skompresowanego elementu specjalne

---

4/ Patrz /9/. W artykule /33/ M. Jackson podaje przykłady zastosowań metod abrewiacji do kodowania nazw zmiennym i stałym w programach komputerowych.

5/ Obszerne kompendium wiedzy z zakresu szyfrowania danych stanowią książki /27/, /28/ oraz /45/ wraz z obszernymi bibliografiami.

6/ Przykłady algorytmów z prawdopodobną utratą informacji, które są stosowane głównie w zakresie kompresji zdjęć i grafiki komputerowej, są zamieszczone w /39/.

7/ Przykład algorytmu dwuprzebiegowego podaje M.F. Lynch /43/. Pierwszy przebieg polega na poprawie wskaźnika zero-jeden /tj. stosunku liczby binarnych zer do jedynek/, a drugi na zastosowaniu algorytmu kompresji długich łańcuchów zer binarnych za pomocą przedrostków o stałej długości.

pole zwane przedrostkiem, które może pełnić różne funkcje:
- licznika usuwanych elementów[8],
- mapy binarnej elementów usuniętych[9] oraz
- etykiety wyróżniającej pola kompresowane od niekompresowanych[10].

Algorytmy obliczeniowe dokonują kompresji danych za pomocą różnych przekształceń algebraicznych wartości kompresowanych elementów. Najczęściej wykonywana jest zmiana podstawy liczenia i zamiana znaków na liczby /25/ i /26/. Aktualnie prace te zostały poważnie rozwinięte przez Rissanena i Langdona /50/. Innym przykładem algorytmu obliczeniowego jest algorytm z użyciem rejestru przesuwającego /ang. Shift register/ proponowany przez Linga i Palermo /41/.

Algorytmy konwersji dokonują zamiany reprezentacji znakowej lub binarnej elementu dziedziny funkcji kompresji na inną reprezentację za pomocą podstawień. W tym celu stosuje się tablice kompresji o różnej wielkości i strukturze. Od prostych podstawień znaku 8-bitowego kodu EBCDIC na znak 5-bitowy w kodzie MKT-2 do złożonych, np. stosowanych w metodzie kompresji bigramów /par znaków/ proponowanej przez Snydermana i Hunta /63/. Do grupy algorytmów konwersji należą również wszystkie odmiany algorytmów kompresji za pomocą kodu Huffmana. Wydaje się jednak celowe wyróżnić tę grupę algorytmów z uwagi na tworzenie kodów o zmiennej długości, problemy określania maksymalnej długości kodu oraz złożony proces dekompresji danych.

W dwóch kolejnych punktach zostaną obecnie przedstawione wybrane algorytmy semantycznych i syntaktycznych metod kompresji danych. Kryterium wyboru była częstość zastosowań algorytmu oraz przydatność dla celów kompresji baz danych[11]. W opracowaniu tym pominięto osobny obszar zastosowań metod kompresji w grafice komputerowej. Tematyce tej poświęcony jest numer 7 czasopisma Proceedings of the IEEE z roku 1980.

Semantyczne metody kompresji danych

Z powodu zapewnienia efektywności procesów przetwarzania komercyjne bazy danych nie są w praktyce jednym łańcuchem znaków lecz zestawem łańcuchów o różnej długości, zwanych rekordami. Każdy rekord posiada strukturę liniową lub hierarchiczną o skończonej liczbie poziomów. Elementami składowymi rekordu są pola elementarne lub złożone, które mogą być jedno lub wielokrotnie powtórzone. Zakres wartości występujących w polu, ich granice maksymalne i minimalne oraz rozkład wartości w poszczególnych wystąpieniach rekordów są bardzo często znane. Wartości takich pól jak: nazwiska, imiona, nazwy dostawców lub odbiorców, nazwy miast, województw itp., mogą być wcześniej znane dzięki badaniom reprezentacyjnym lub całościowym zbiorów danych funkcjonujących w tradycyjnej technologii przetwarzania danych. Badania takie można prowadzić w sposób ręczny lub zautomatyzowany dzięki użyciu specjalnego oprogramowania[12].

W rezultacie wartościom najczęściej występującym można przydzielić kody, które będą wstawiane do bazy danych. Przykładem mogą tu być badania prowadzone przez J.Martina a dotyczące imion obywateli USA, które z racji różnych narodowości ich obywateli charakteryzują się szczególnie dużą różnorodnością[13]. Wyniki były zaskakujące, ponieważ okazało się, że 90% obywateli USA używa tylko 256 imion. Można było zatem przydzielić dla najczęściej używanych imion jednobajtowy kod, z którego wydzielono 1 kombinację poprzedzającą właściwe imię jednego z pozostałych 10% obywateli. Opisane rozwiązanie wymaga utrzymania w pamięci operacyjnej tablicy o rozmiarze 5355 bajtów /1 bajt kodu plus 20 bajtów na imię razy 255/. Efektem tego rozwiązania jest kompresja rzędu 86%.

Inny przykład kompresji danych dominujących przedstawia P.Alsberg /3/. Z uwagi na nieefektywność kompresowania wszystkich wartości danych zaleca on badanie częstości występowania wartości danych. Badając nazwy czterech tysięcy dostawców części oraz dostarczane przez nich materiały ustalił, że 80% części jest zamawianych u 48 dostawców. W wyniku tego zaproponował 6-bitowe pole stałej długości dla najważniejszych dostawców, w którym wartości od Ø do 47 adresują ich pełne nazwy w tablicy kompresji, natomiast wartości od 48 do 63 świadczą o wystąpieniu dodatkowych 8 bitów. Tak opisane pole 14-bitowe umożliwia przecho-

---

8/ Przykładem algorytmy usuwające spacje /46/ i /54/.

9/ Patrz /5/ oraz /75/.

10/ Przykładem algorytm kompresji stosowany w SZBD IDMS firmy Cullinane corp. /13/.

11/ Obszerniejsze wykłady na temat kompresji danych znajdują się w opracowaniach /54/, /75/, /5/, /49/ oraz /46/. Algorytmy, które uznano za szczególnie przydatne dla celów kompresji baz danych zaznaczono na rysunku 1 pogrubioną ramką.

12/ Opracowany w Instytucie Informatyki AE we Wrocławiu program automatycznej analizy struktur zbiorów danych - WAD, umożliwia aktualnie zebranie danych statystycznych niezbędnych do prawidłowego projektowania większości znanych metod kompresji /38/.

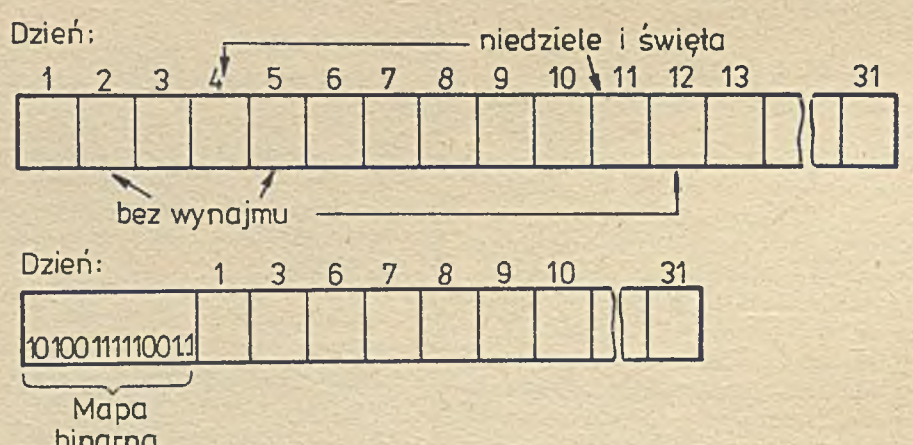
13/ Patrz /46 s. 489/.

wanie na pierwszych 6 bitach 16 adresów tablic dostawców, a pozostałych 8 bitów jest adresem względnym pozycji dostawcy w każdej tablicy. Istnieją zatem dwa typy pól o długości 6 lub 14 bitów i średniej długości /6x0,8/ + /14x0,2/= 7,6 bita na znak. Zakładając, że źródłowa wartość pola DOSTAWCA miała długość 24 bajtów oczekiwana wartość współczynnika kompresji będzie wynosić 96%.

Na marginesie przedstawionego rozwiązania oraz uzyskanego współczynnika kompresji nasuwa się jedna uwaga. Kodowanie wartości dominujących za pomocą kodów binarnych o stałej lub zmiennej długości wymaga tworzenia i przechowywania dużych tablic kompresji dla każdego kompresowanego pola. W opisanym przykładzie kompresji pola DOSTAWCY wymagany obszar dla tablicy wynosi 104 000 bajtów /24 nazwa + 2 adres x 4000 dostawców/. Umieszczenie tak dużej tablicy w pamięci operacyjnej jest prawie niemożliwe dla większości komputerów używanych w naszym kraju do przetwarzania danych gospodarczych, a nieefektywne dla pozostałych komputerów. Znając częstość występowania nazw dostawców możemy w pamięci operacyjnej przechowywać jedynie jedną tablicę najważniejszych dostawców o wielkości 1248 bajtów /26x48/. Pozostałych 16 tablic będzie zorganizowanych w postaci zbioru o bezpośrednim dostępie posiadających 16 rekordów stałej długości 6656 bajtów, zawierających 256-krotne pole złożone z nazwy i 14-bitowego kodu. Opisane rozwiązanie będzie ponadto posiadało 144 wolne pozycje adresowe przeznaczone na zakodowanie nowych dostawców.

Bardzo często w strukturze rekordu rezerwuje się miejsca na pola, których wystąpienie jest uzależnione od innych okoliczności. Podobne zjawisko występuje w polach wielokrotnych zawierających wartości zerowe dla pozycji, które nie zawsze muszą wystąpić. Dobrą metodą kompresji dla opisanych przypadków jest mapa binarna użyta na poziomie rekordu lub pola wielokrotnego. Na rysunku 2 przedstawiono przykład struktury rekordu, w którym rejestrowane są wpływy dzienne z tytułu wynajmu samochodów w punktach wynajmu. Z uwagi na fakt, że niektóre punkty nie działają w niedziele i święta oraz że niecodziennie dochodzi do wynajmu, zdecydowano się na zastosowanie map binarnych dla pola wielokrotnego WYNAJMY-W-M-CU. Każde pole elementarne rejestrujące wpływy dzienne posiada długość pięciu bajtów. Po kompresji tworzona jest na początku tablicy jednosłowowa mapa binarna, która umożliwia ustawienie 32 wskaźników bitowych /1 binarna oznacza niezerową wartość pola/. Dla przykładowych danych z rysunku 2 możliwa wartość kompresji wynosi /dziesięć pól niezerowych/ 65%.

Przykładem zastosowania mapy binarnej do oznaczania wystąpień pól lub podpól w polach wielokrotnych może być fragment rekordu charakterystyki pracownika zawierający opis ostatnich trzech miejsc pracy wraz z opinią. Każde z podpól posiada zmienną długość od 0 do 23 wierszy 80-bajtowych. Jeśli pracownik od początku pracuje w macierzystym zakładzie to dla wystąpienia jego rekordu opisane pole nie występuje, co jest oznaczone za pomocą trzech dwubajtowych pól liczników wierszy, każdy o wartości równej zero. Użycie jednobajtowej mapy binarnej umożliwia adresowanie do 8 podpól. Jeśli pole występuje to jego pierwsze dwa bajty będzie stanowił licznik długości /półsłowo binarne/, a kompresja wyniesie 83% i jako efekt dodatkowy pozwoli skrócić proces rozpoznania struktury i wielkości pola wielokrotnego z warunkowym występowaniem wartości podpól.



Rys. 2. Przykład zastosowania mapy binarnej do usuwania zbędnych podpól pola wielokrotnego WYNAJMY - W-M-CU

Przedstawiona uprzednio metoda kodowania wartości dominujących w tablicach kompresji pól znajduje swoje zastosowanie również przy przechowywaniu różnego rodzaju krótkich tekstów stałych dla określonej bazy danych. Przykładem mogą być:
- nazwy kooperantów, dostawców i odbiorców,
- komunikaty diagnostyczne i ostrzegawcze w informacyjnych sieciach informatycznych,
- nazwy wyrobów, części, narzędzi, itp.

Tworzenie dla każdego pola osobnej tablicy kompresji w większości przypadków będzie obciążone nadmierną redundancją. Wynika to z faktu, że dostawcy często są również odbiorcami i występuje powtórzenie całej nazwy lub kilku jej członów, jeśli kooperantami jest kilka zakładów należących do tego samego zrzeszenia /np. kopalnie, huty/, rozmieszczone na terenie jednego regionu /np. Dolnośląskie Zakłady Przemysłu .../, należące do sektora gospodarki spółdzielczej /np. Spółdzielnia Pracy Inwalidów ..., Okręgowa Spółdzielnia Mleczarska .../, itp. Komputerowa analiza tych nazw pozwoli na wyodrębnienie wszystkich słów wraz z częstościami ich występowania. Dysponując także danymi odnośnie częstości kontaktów z każdym kontrahentem możemy zbudować jedną wspólną tablicę kompresji członów nazw lub skorowidz indeksujący różne rekordy i pola w różnych zbiorach. Przykład rozwiązania podobnego problemu dla tablic diagnostycznych w kompilatorze języka PL/C podaje R. Wagner /66/. Rozwiązanie, w którym zastosowano skorowidz najczęstszych słów w danych tekstowych można znaleźć w artykule S.F.Weiss'a i R.L. Vernora /70/. Z uwagi na odmienność zasad nazewnictwa jednostek gospodarczych w celu zastosowania opisanej metody należałoby poprzedzić dokładną analizą wykazu nazw jednostek gospodarczych w PRL.

Podobnie jak nazwy firm można dokonać kompresji nazwisk i imion. W zbiorach o postaci źródłowej nazwiska i imiona są najczęściej umieszczane w polach o stałej długości 30 znaków. Badania prowadzone na zbiorze nazwisk angielskich przez W.Nugent'a /47/ oraz J. Dolby'ego /15/ nie doprowadziły do uzyskania efektywniejszego algorytmu od znanych algorytmów dla danych tekstowych. Ubocznym efektem tych prac była ocena przydatności różnych algorytmów kodowania nazwisk, które umożliwiają neutralizowanie błędów pisowni lub wymowy w systemach rezerwacji miejsc lub systemach informacji bibliotecznej[14]. Z uwagi na generowanie synonimów wspomniane kody mogą mieć znaczenie pomocnicze polegające na ułatwieniu procesów wyszukiwania według tak powszechnego identyfikatora, jak nazwisko lub nazwisko z inicjałami.

W komercyjnych bazach danych istnieje wiele wartości pól, nazw i tekstów komunikatów, dla których opisane wcześniej metody kompresji dają gorsze rezultaty niż zastosowanie metod specjalnych. Jednym z przykładów jest pole daty, które w większości systemów informatycznych przedstawiane jest w postaci 6 znaków. Wybierając arbitralnie jakąś datę jako punkt odniesienia możemy przedstawić dowolną datę w postaci przyrostu dni od daty pierwotnej. Opisany sposób przedstawiania daty w kalendarzu Juliańskim został zastosowany przez Jose Skaligero w 1582 roku do oznaczania wystąpień zjawisk astronomicznych[15]. Użycie kodu 22-bitowego umożliwia datowanie zjawisk do roku 6700. Krótszy kod 16-bitowy, wystarczający dla większości zastosowań, umożliwia datowanie w przedziale 175 lat. Zastosowanie opisanej metody przedstawiania daty zostało implementowane w systemie komputerowym ODRA 1300, w którym data jest przechowywana w jednym słowie /24-bitowym/ w postaci przyrostu dni od dnia 31.12.1899 roku.

Różne sposoby prezentacji daty dla zjawisk gospodarczych mogących występować w okresie 20 lat prezentuje P.A. Alsberg /3/. Rozważa on argumenty za i przeciw przechowywaniu daty w postaci binarnej liczby całkowitej oraz trzech skonkatenowanych pól binarnych: dnia, miesiąca i roku. Ponadto podaje inne przykłady kompresji pól nietypowych jak np. płeć.

Większość rozważań na temat kompresji pól nietypowych do postaci pól binarnych o stałej lub zmiennej długości abstrahuje od faktu, że operowanie tak skompresowaną strukturą rekordu zmiennej długości nie jest możliwe we wszystkich językach programowania, a w większości przypadków powoduje straty w postaci dodatkowych bitów lub bajtów synchronizujących pola binarne.

Syntaktyczne metody kompresji danych

Syntaktyczne metody kompresji danych działają na zbiorze elementów danych, traktując je jako jedno lub wieloelementowe łańcuchy o stałej lub zmiennej długości. Długość łańcucha może być jednością lub krotnością podstawowych elementów fizycznej lub logicznej struktury danych. Do najczęściej używanych elementów należą: bit lub n-tki bitów, bajt lub n-tki bajtów, słowo tekstu lub fraza[16]. Przyjmując za kryterium klasyfikacji typ łańcucha możemy wyróżnić metody kompresji łańcuchów: bitowych, bajtowych /znakowych/, frazowych lub słów.

14/ Do tego celu szczególnie użyteczne są kody stosujące algorytmy: SOUNDEK, ALPHACHECK, Transition Distance Code /47/.

15/ Patrz /46 s.468/.

16/ N-tki bajtów /znaków/ określane są również mianem n-gramów /19/ i /44/ lub X-gramów /68/.

Struktury bitowe są zazwyczaj tworzone dla pewnych specyficznych struktur danych. Przeważnie stosuje się je do kodowania obrazów lub tworzenia wektorów wskaźników łączników adresowych w skorowidzach baz danych [35]. W obu przypadkach dane mają postać struktury binarnej liczącej kilkadziesiąt tysięcy bitów /w skorowidzach/ lub kilka milionów bitów /cyfrowa prezentacja obrazów/. Częstość występowania zer znacznie przekracza częstość jedynek, a ponadto te ostatnie przejawiają silną tendencję do grupowania się.

W literaturze opisano wiele algorytmów kompresji nieprzerwanych łańcuchów binarnych zer zakończonych jedynką /ang. run/, że wymienimy tu prace /20/, /10/, /35/, /37/, /7/, /64/ oraz /39/. Jeśli przyjmiemy, że d oznacza maksymalną długość łańcucha zer kodowanego jednym słowem kodowym /np. 8 lub 16/ to przykładowa kompresja łańcucha bitowego za pomocą algorytmu Bradley'a realizowana jest następująco. Słowa kodowe o stałej długości są podzielone na dwie grupy. Do pierwszej grupy należą te, które będą symbolizowały łańcuchy z przedziału /1, d/. Jeśli długość słowa będzie oznaczona przez k, to do drugiej grupy będą należały ciągi kodowe odpowiadające nieprzerwanym łańcuchom zer o długościach d, 2d, ..., 2^k-d, co przedstawia poniższa tablica

Przykład zastosowania kompresji łańcucha bitowego za pomocą algorytmu Bradley'a

Tablica kodu Bradley'a

	Długość łańcucha	Słowo kodowe
I grupa	1	000
	2	001
	3	010
	4	011
	5	100
II grupa	5	101
	10	110
	15	111

przed kompresją:
000000010000001000001001 24 bity
po kompresji:
100011010011010101 18 bitów
współczynnik kompresji $\frac{24-18}{24} \cdot 100\% = 25\%$

Większość opisanych w literaturze metod kompresji dotyczy łańcuchów znakowych. Kompresja łańcucha znaków bazy danych można wykonać w całości lub dzieląc go na podłańcuchy o dowolnie ustalonej długości. Generalnie procesy kompresji dekompresji są szybsze dla krótszych łańcuchów, ale z drugiej strony kompresja łańcuchów dłuższych pozwala na uzyskanie lepszego współczynnika kompresji. Długości kompresowanych łańcuchów mogą zawierać się w bardzo szerokim przedziale /od jednego do kilku tysięcy znaków/, stąd ich prezentację rozpoczniemy od kompresji pojedynczych znaków. Kompresja pojedynczych znaków może być realizowana w prosty sposób dzięki zmianie używanego kodu. Powszechnie stosowany kod EBCDIC pozwala na zakodowanie 256 różnych typów znaków.

W praktyce wiele zbiorów danych zbudowanych jest z nie więcej niż 64 typów alfanumerycznych znaków. W tym zestawie mieszczą się wszystkie znaki alfanumeryczne i specjalne. Stosując dla każdego znaku kod 6-bitowy /podobny jak w komputerach serii ODRA 1300/ uzyskamy zapis zawartości słowa maszynowego w trzech bajtach, a w rezultacie kompresję o współczynniku 25% i bardzo prostym oraz łatwym w użyciu algorytmie [18].

Jeśli zbiór danych zawiera wyłącznie znaki alfabetyczne, numeryczne i ograniczony zestaw znaków specjalnych, to do reprezentacji danych można zastosować 5-bitowy kod telegraficzny nr 2. W razie jego użycia można lokować po 8 znaków w pięciu bajtach pamięci i po niewielkich modyfikacjach można uzyskać kompresję rzędu 37% [19].

Jeżeli zbiór danych zawiera wyłącznie znaki numeryczne to w prosty sposób można uzyskać kompresję rzędu 50% [20]. Niektóre firmy komputerowe umożliwiają sprzętową realizację tak efektywnego przechowywania danych numerycznych [21].

Większe efekty kompresji pozwalają uzyskać kody zamieniające łańcuchy znaków w jeden, najczęściej nieużywany, znak kodu EBCDIC. Przykład kodu z zastosowaniem n-gramu dwuznakowego /bigramu/ opisuje Snyderman i Hunt /63/. Dzięki badaniom statystycznym tekstów naturalnego języka angielskiego wyodrębnili oni trzy grupy znaków. Pierwszą grupę zawierającą osiem znaków, nazwali

---

17/ Badania prowadzone przez M.F.Lyncha /43/ wykazały znikomy efekt kompresji łańcuchów bitowych dla baz danych bibliograficznych. Dopiero dodatkowa modyfikacja "poprawiająca" stosunek zer do jedynek /od 52 do 119%/ pozwoliła uzyskać kompresję rzędu 22,1 - 31.7%.

18/ Patrz /1/.

19/ Patrz /46 s.493-495/.

20/ Patrz /31/, /12/ oraz /62/.

21/ Dotyczy to na przykład dużych komputerów System/370 firmy IBM /31/.

"znakami głównymi", drugą "znakami dołączanymi", a trzecią grupę "znakami niedołączanymi", wykaz znaków każdej grupy zawiera rysunek 3.. Dzięki temu, że w kompresowanym tekście użyto tylko 88 znaków, pozostałych 168 znaków może reprezentować różne bigramy, których wybrane przyklady zawierają ostatnie kolumny tablicy z rysunku 3. Procedura kompresji jest następująca:

Jeśli początkowy znak kompresowanego łańcucha nie jest "znakiem głównym", to jego kod pozostaje niezmieniony inaczej jeżeli następny znak nie jest "znakiem dołączanym", to jego kod również pozostaje niezmieniony, inaczej oba znaki zostają zastąpione nowym kodem, którego wartość jest sumą kodów składowych. Proces jest kontynuowany dla pozostałych znaków łańcucha.

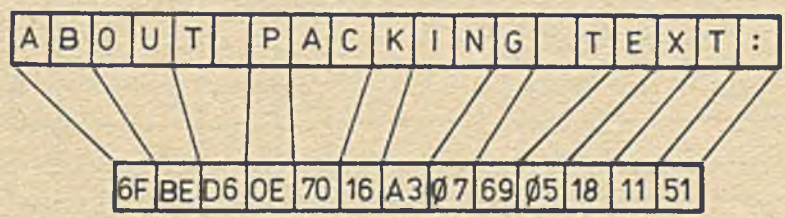
Stosując opisany kod Snyderman i Hunt osiągnęli współczynnik kompresji równy 35%. Dzięki dokładniejszej analizie tekstu i odmiennie ustalonych bigramach Schie i Thomas [55] osiągnęli kompresję o wartości 43,5%[22].

Podsumowując charakterystyki metod kompresji łańcuchów znakowych za pomocą bigramów można stwierdzić, że są one proste w użyciu i efektywne dla większości danych tekstowych.

22/ Stosując podobną metodę G.C.Jewell [34] osiągnął współczynnik kompresji rzędu 47%, a ponadto przyspieszył proces dekompresji dzięki adresowanej organizacji tablicy bigramów. Model oceny efektywności metod kompresji za pomocą bigramów jest zawarty w pracy [8].

Znaki główne		Znaki dołączane		Znaki niedołączane						Pary znaków			
Znak	HEX	Znak	HEX	Znak	HEX	Znak	HEX	Znak	HEX	Znak	HEX	Znak	HEX
Ø	58	Ø	ØØ	J	15	q	2B	<	41	ØØ	58	AØ	6D
A	6D	A	Ø1	K	16	r	2C	(	42	ØA	59	AA	6E
E	92	B	Ø2	Q	17	s	2D	+	43	ØB	5A	AB	6F
I	97	C	Ø3	X	18	t	2E	&	44	ØC	5B	AC	7Ø
O	AC	D	Ø4	Y	19	u	2F	!	45	ØD	5C	..	..
N	C1	E	Ø5	Z	1A	v	3Ø	$	46	ØE	5D	AW	81
T	D6	F	Ø6	a	1B	w	31	*	47	ØF	5E	EØ	82
U	E8	G	Ø7	b	1C	x	32	)	48	ØG	5F	EA	83
		H	Ø8	c	1D	y	33	;	49	ØH	6Ø	..	..
		I	Ø9	d	1E	z	34	-	4A	ØI	61	EW	9C
		L	ØA	e	1F	Ø	35	/	4B	ØL	62	IØ	97
		M	ØB	f	2Ø	1	36	,	4C	ØM	63	..	..
		N	ØC	g	21	2	37	%	4D	ØN	64	OØ	AC
		O	ØD	h	22	3	38	_	4E	ØO	65	..	..
		P	ØE	i	23	4	39	>	4F	ØP	66	NØ	C1
		R	ØF	j	24	5	3A	?	5Ø	ØR	67	..	..
		S	1Ø	k	25	6	3B	:	51	ØS	68	TØ	D6
		T	11	l	26	7	3C	#	52	ØT	69	..	..
		U	12	m	27	8	3D	@	53	ØU	6A	UØ	E8
		V	13	n	28	9	3E	'	54	ØV	6B	..	..
		W	14	o	29	¢	3F	=	55	ØW	6C	VW	FF
				p	2A	.	4Ø	"	56				
								¬	57				

Przykład kompresji:



Rys. 3. Tablica kodu Snydermana i Hunta wraz z przykładem

Maksymalną wartość współczynnika kompresji wynoszącą 50%, można uzyskać wówczas, gdy każdy znak kompresowanego tekstu będzie wchodził w skład wybranych bigramów.

Komercyjne bazy danych, zbiory danych i komunikaty przesłane w sieciach telekomunikacyjnych zawierają bardzo często jeden, dwa lub kilka znaków o szczególnie dużej częstości występowania. Cytowany już wielokrotnie J. Martin /46/ prezentuje przykłady baz danych, w których udział zer wynosi 55,5% a spacji 1,1%. Badania prowadzone przez Lynch'a, Petrie oraz Snella /44/ na różnych tekstowych bazach danych w ciągu trzech lat wskazują na około 15% udział spacji. Istnieje wiele efektywnych metod usuwania znaków najczęstszych, występujących zarówno w krótkich /do trzech/ jak i w długich /powyżej trzech/ łańcuchach znaków powtarzalnych. Przyjmijmy dla dalszych rozważań, że w hipotetycznej bazie danych występuje znak odstępu /spacja/, którego częstość występowania wynosi 0,5 lub inaczej 50%. Wartość częstości występowania znaku obrazuje dobrze częstość występowania danego typu znaku, nie dostarcza jednak informacji o wystąpieniach n-tek znaków tego samego typu /ich liczebności, rozkładzie i innych parametrach statystycznych/. Odmienne metody należy przyjąć dla sytuacji występowania co drugiego znaku spacji w bazie danych /łącznie będzie to również udział 50%/ a inne wówczas, gdy rozkład częstości n-tek spacji będzie zbliżony do rozkładu normalnego lub Poissona. Zostaną teraz przedstawione metody kompresji znaków powtarzalnych występujących w krótkich łańcuchach /tj. takich n-tkach, gdzie n<3/.

Pierwszą z metod jest metoda map /masek/ binarnych, która polega na wprowadzaniu na początek kompresowanego łańcucha /rekordu/ mapy binarnej, odpowiedniej do długości rekordu, w której dla każdej pozycji znakowej /niezerowej/ będzie ustawiony bit /B_1=1/, a wartości zerowe będą prezentowane w mapie przez binarne zero na odpowiedniej pozycji, co ilustruje rysunek 4. Technika ta użyteczna jest dla rekordów stałej i zmiennej długości umieszczanych w pamięci, której komórkami podstawowymi są bajty lub słowa. Wadą metody są dodatkowe narzuty zajętości pamięci z tytułu przechowywania map binarnych.

Drugą z metod kompresji krótkich łańcuchów znaków jest metoda określona w niniejszym artykule nazwą ZSP-KOD[23]. Metodę tę można stosować jedynie dla baz danych, w których nie występują znaki małych liter /tzw. znaki dolne na klawiaturze/. W tym przypadku wszystkie dane w bazie posiadają drugi bit o

23/ Patrz /46 s.486 i 487/.

Przed kompresją:

Kod transakcji	Symbol wyrobu	JM	Ilość zamawiana		itd. /razem 80 znaków/
ØØ7	ØØ482Ø39Ø	3Ø	ØØØØØØ9ØØØØ		

Po kompresji:

Mapa binarna /10 bajtów = 80 bitów/				
ØØ1ØØ111Ø11Ø1ØØØØØØØ1ØØØØ... ØØØ	7	48239	3	9

Rys. 4. Kompresja zbytecznych zer z użyciem mapy binarnej

wartości = 1. Korzystając z tego faktu można ustawić drugi bit na Ø, co oznacza kompresję zbędnych znaków. Pozostałych 7 bitów będzie wyznaczało zarówno typ usuwanych znaków jak też ich długość /maksymalnie 64 znaki/. co ilustruje rysunek 5.

```
Ø - oznacza usuwanie np. zer
1 - oznacza usuwanie np. spacji
X Ø X X X X X X
        sześć bitów podaje liczbę usuwanych
        znaków /maks. 64/
Ø - oznacza kompresję
1 - brak kompresji /tzn. znak danej/
```

Rys. 5. Kompresja zbytecznych zer i spacji z użyciem wybranych wartości kodu EBCDIC.

Jeśli usuwamy znaki tylko jednego typu, to nie trzeba tego faktu specjalnie deklarować. Metoda ZSP-KOD jest zbliżona w efektywności do metody map binarnych, ale ponadto umożliwia kompresję dwóch typów znaków. Wstępne badania wykonane za pomocą prototypu programu analizy procesów kompresji danych-KOMPRAN wykazały, że obie metody są sprawniejsze od najbardziej rozpowszechnionej metody ogranicznika z licznikiem /ZSP-LICZ/. Dla rozkładu n-tek spacji zbliżonym do rozkładu Poissona oraz częstości występowania spacji wynoszącej 0.555 uzyskano następujące współczynniki kompresji: ZNAK-MAP - 38,75%, ZSP-KOD-41,94%, ZSP-LICZ-23,62%. Wyraźnie niższa wartość współczynnika kompresji metody ZSP-LICZ wynika z faktu jej niewłaściwego zastosowania. Jest to bowiem bardzo efektywna i prosta metoda, ale dla długich łańcuchów powtarzalnych znaków. W celu kompresji łańcucha stosuje wybrane znaki specjalne /nieużywane w danych źródłowych/, które symbolizują kompresję zer lub spacji oraz liczniki usuniętych, sąsiadujących ze sobą znaków jednego rodzaju. Ilustracją tej metody jest rys. 6.

```
Dane źródłowe: JWW1ØØØØTK   28ØØØØ
Dane skompresowane: JWW1%4TK$528%4
        gdzie: % - etykietuje usuwane zera,
               $ - etykietuje usuwane spacje.
```

Rys. 6. Kompresja zbytecznych zer i spacji z użyciem znaków specjalnych /%,$/ i liczników długości.

Jeśli w danych źródłowych mogą pojawić się wszystkie znaki /np. w kodzie sześciobitowym/ to należy wybrać dwa o najmniejszej częstości występowania i ich obecność w tekście /w charakterze danych/ zdublować pozostawiając pojedyncze wystąpienia jako etykiety usunięć znaków zbędnych. Opisana technika była wielokrotnie stosowana w praktyce, przyczyniając się do uzyskania kompresji od 24 do 61%[24], a w niektórych przypadkach nawet 63 do 75%[25]. Zastosowanie odmiany tej metody do kompresji bibliotek programów źródłowych opisuje B. Hahn [26], a do kompresji danych tekstowych R. Fajman oraz J. Borgelt, opisując pakiet redagowania tekstów WYLBUR [18].

Zmieniając w rekordach bibliotek programów źródłowych typ rekordu, ze stałej na zmienną długość, usuwając spacje poprzedzające i następujące po instrukcji programu oraz zmieniając ośmioznakowy identyfikator numeru wiersza na dwubajtowy, pakiet programowy obsługi bibliotek źródłowych LIBRARIAN pozwala na przechowywanie jednego wiersza przeciętnie na 25 bajtach pamięci /czyli kompresja wynosi prawie 70%/ [4].

Opisane tu metody usuwania zbytecznych zer i spacji są proste w użyciu i godne upowszechniania. Wiadomo także, że ich możliwości kompresji są mniejsze od najefektywniejszej z metod, tj. kodu Huffmana. Wyniki badań analitycznych prowadzonych za pomocą programu KOMPRAN zostały przedstawione w tablicy 2.

Kody o zmiennej długości stosowane w procesach kompresji danych bazują najczęściej na liczącym ponad 30 lat algorytmie D. Huffmana [29], który zajmował się między innymi badaniami kodów o minimalnej redundancji[26]. Kody te są budowane na podstawie obserwacji o nierównomiernej częstości występowania kodowanych elementów w zbiorze, co dobrze ilustruje tablica 1. Algorytm tworzy tablicę konwersji elementów na łańcuchy bitowe o zmiennej długości. Element najczęstszy jest reprezentowany przez łańcuch najkrótszy a element najmniej częsty przez łańcuch o względnie większej liczbie bitów. Każdy łańcuch bitów jest jednoznacznie dekodowalny a całość pozwala uzyskać kod charakteryzujący się najmniejszą liczbą bitów reprezentujących element. Przykładowy kod dla danych z tablicy 1 zawiera tablica 3.

W praktyce zastosowanie kodu Huffmana pozwalało na kompresję danych rzędu co najmniej 50% [46] i [32][27]. Pomimo tak dużej kompre-

24/ Patrz [14].

25/ Patrz [4].

26/ Szczegóły budowy kodów Huffmana można znaleźć w prawie każdym podręczniku z zakresu teorii informacji i kodowania, np. [2], [53], [60].

27/ Zastosowanie kodów Huffmana w systemach minikomputerowych opisuje [48].

Tabela 2

Porównanie oczekiwanych wyników kompresji małego zbioru danych gospodarczych

Analiza metod kompresji zbioru danych
T028JWW.JAN.FORT(AEKARTY)
zawierajacego 947 rekordow o dlugosci stalej 80 bajtow kazdy.
W analizowanym zbiorze wystepuje 38 typow znakow kodu EBCDIC. Obszar wynosi 76 k-bajtow a koszt przechowywania na dobe 5,2 zlotych.

Nazwa procedury	Wsp. kompresji (XX)	Obszar zbioru po kompresji (K-bajty)	Koszt przech. zbioru na dobe (zl)	Szacowany czas kompresji tysiaca znakow (w sek.)	Szacowany czas kompresji calego zbioru (w min)	Szacowany czas dekompresji tysiaca znakow (w sek.)	Szacowany czas dekompresji calego zbioru (w min)
HUFF8	75.00	18.94	1.3	0.1344	0.0424	0.3782	0.1194
HUFF	-	-	-	-	-	-	-
BIT5	31.44	51.94	3.6	0.0780	0.0675	0.0700	0.0606
BIT6	25.00	56.82	4.0	0.0220	0.0208	0.0180	0.0170
KOD13-3	45.93	40.95	2.9	0.0730	0.0498	0.0650	0.0444
HAHN	33.33	50.51	3.5	0.1199	0.1009	0.3148	0.2650
ZSP-LICZ	23.62	57.87	4.1	0.0320	0.0309	0.0240	0.0231
ZSP-KOD	41.94	43.98	3.1	0.0390	0.0286	0.0280	0.0205
ZNAK-MAP	47.50	39.77	2.8	0.0360	0.0239	0.0400	0.0265

sji nie jest to algorytm szeroko stosowany z uwagi na:
- duży stopień złożoności algorytmu,
- stosunkowo duży narzut czasu kompresji nawet w dużych komputerach,
- problemy synchronizacji rekordów i rozwiązanie zagadnienia dekompresji ostatniego bajtu,
- dużą wrażliwość kodu na zmiany częstości występowania znaków,
- problemy implementacji wynikające z maksymalnej długości słowa kodowego o zmiennej długości.

Tablica 3

Przykład zastosowania kodu Huffmana o zmiennej długości /dla danych z tablicy 1/

i	Kodowany znak	Częstość /pi/	Wartość kodu	Długość kodu /li/
1	odstęp	58782	1	1
2	0	12558	00	2
3	1	04868	0100	4
4	2	03478	0110	4
5	4	01913	01111	5
6	3	01715	01110	5
7	5	01543	010111	6
8	6	01535	010110	6
9	7	01213	010101	6
10	8	01210	010100	6

Dwa pierwsze problemy będą zapewne efektywnie rozwiązane w najbliższym czasie dzięki mikroprogramowej realizacji algorytmu w jednostce centralnej lub zbudowaniu mikroprocesorowych przystawek. W odniesieniu do struktur danych charakterystycznych dla baz danych można stwierdzić, że są one mniej podatne na uszkodzenia z tytułu zagubienia bitu. Powszechnie stosowane liczniki długości rekordów oraz techniki dekompresji ostatniego bajtu, w którym trzeba rozpoznać skrajne lewe bity a zignorować pozostałe, sprawiają, że obecnie nie jest to tak istotny problem jak przykładowo w telekomunikacji[28].

Istotne zmiany częstości występowania znaków mają bardzo duży wpływ na efektywność kodu Huffmana. Jednak w odniesieniu do głównych zbiorów lub baz danych różnice te są niewielkie [44]. Aby uniknąć drastycznych różnic wartości uzyskiwanego współczynnika kompresji należy wprowadzić monitorowanie wyników kompresji po każdej aktualizacji lub "poprawiać" skośność rozkładu częstości znaków, np. przez niekompresowanie pól binarnych lub tworzenie osobnych tablic dla zbiorów transakcji poszczególnych typów.

Jak dotychczas nie jest rozwiązany problem obliczenia maksymalnej liczby bitów w słowie kodowym. Istnieją natomiast wzory na obliczenia granicy maksymalnej liczby bitów [42], która wynosi:

28/ Techniki dekompresji ostatniego bajtu zostały opisane w [42] i [21]. Autor programu kompresji metodą Huffmana mgr inż. Jacek Korta zastosował jeszcze inną technikę polegającą na pamiętaniu w skompresowanym rekordzie /w półsłowie/ liczby bajtów rekordu źródłowego. Algorytm ten wchodzi w skład przygotowywanego pakietu kompresji danych -KOMPRES.

$$\lceil(-\log_2 p_i)\rceil$$

gdzie:
pi - jest najmniej częstym znakiem w zbiorze źródłowym a znaki $\lceil\ \rceil$ reprezentują funkcję zaokrąglania wyniku "w górę" do wartości całkowitej.

Ponieważ wiadomo, że wiele znaków alfabetu źródłowego może wystąpić, ale w trakcie badań statystycznych ich częstość była zerowa, to w tablicy kompresji muszą one posiadać odpowiednie słowa kodowe. Jeśli zatem pi będzie w dalszym ciągu reprezentować najmniejszą niezerową częstość znaku, a k będzie liczbą znaków w alfabecie źródłowym o zerowej częstości występowania, to granica maksymalnej liczby bitów słowa kodowego będzie wynosić:

$$1 + \lceil(-\log_2 p_i)\rceil + \lceil\log_2 k\rceil$$

Dla zbioru danych o częstościach znaków przedstawionych w tablicy 1 granice te będą wynosiły odpowiednio 17 i 26. Istnieje wiele sposobów ograniczenia maksymalnej długości słowa kodowego, z których kilka jest zasygnalizowanych w artykułach /42/ oraz /22/.

Pewną uproszczoną wersją kodów zmiennej długości są kody dwupoziomowe, które dzielą zbiór znaków alfabetu źródłowego na dwie części. Pierwsza zawiera 2^{k-1} najczęstszych symboli, a druga resztę. Jeżeli przyjmiemy, że liczba znaków alfabetu źródłowego n jest równa 2^J to minimalizując wyrażenie /dla k=2 do 7/

$$K + J - (J*P/k),$$

otrzymamy średnią liczbę bitów na znak. Utworzony zostanie kod o k bitach, który będzie reprezentował 2^{k-1} najczęstszych znaków, zostawiając jednocześnie ostatnią kombinację na przedrostek poprzedzający znaki drugiej grupy. Przykładem kodu dwupoziomowego jest kod, który będzie określany nazwą KOD13-3. Nazwa ta wynika z przyjętej zasady kodowania wybierającej 13 najczęściej występujących znaków, przydzielając im czterobitowe wartości binarne od ØØØØ do 11ØØ. Kolejne wartości binarne, tj. 11Ø1, 111Ø oraz 1111 stanowią odsyłacze do trzech tablic zawierających pozostałe znaki o mniejszej częstości występowania. Jeżeli przeczytane 4 bity mają wartości 11Ø1, 111Ø lub 1111 to następne stanowią jednocześnie odsyłacz do jednej z trzech tablic oraz względny numer znaku w tablicy. W przypadku, gdy pierwsze 13 znaków reprezentuje 80% wystąpień znaków w łańcuchu, to średnia liczba bitów na znak wynosi 4,8, a współczynnik kompresji wynosi 40%.

W artykule przedstawiono wiele powszechnie stosowanych metod kompresji danych. Z uwagi na ścisłe sprecyzowanie tematu nie zaprezentowano metod kompresji, które są charakterystyczne dla innych zastosowań, np. kody przyrostowe w telemetrii /65/ lub kompresja wzdłużna w telekomunikacji. Z uwagi na rozwój oprogramowania w naszym kraju mini i mikrokomputerów należy rozważyć możliwość zastosowania różnych metod kompresji nie tylko dla baz danych, ale również systemów operacyjnych, pakietów redagowania tekstów, pakietów obsługi bibliotek programów, programów składowania danych, itp. Jednocześnie należy rozwinąć badania nad możliwością sprzętowej realizacji algorytmów kompresji lub w połączeniu z algorytmami szyfrowania bądź wyszukiwania. Sytuacja naszych użytkowników jest bowiem odmienna od sytuacji w ośrodkach obliczeniowych krajów bogatych, w których nabycie dodatkowych pakietów, jednostek dyskowych lub komputerów jest sprawą tylko posiadanych środków finansowych.

Literatura

/1/ S.G.Abbey, J.Geller, A.D'Enrico: Development and Optimization of a Text Compression/Decompression Algorithm. Proc. COMPSAC 79, THE IEEE Conference, Chicago 1979, s. 618-622.

/2/ N.Abramson: Teoria Informacji i Kodowania. /Tłum. z ang./, PWN Warszawa 1969, s. 222.

/3/ P.A.Alsberg: Space and Time Savings Through Large Data Base Compression and Dynamic Restructurins, Proc. IEEE, 63, 8 August 1975, s. 1114-1122.

/4/ Applied Data Research, Inc. The Librarian. Route 206 Center, CN Princetown, New Jersey 08540.

/5/ J.Aronson: Data Compression - a Comparison of Methods. National Bureau of Standards Special Publication 500-12, June 1977, s.39.

/6/ R.B.Arps: Bibliography on Binary Image Compression. Proc. IEEE, Vol. 68, 1980, s. 922-924.

/7/ L.R. Bahl, H.Kobayashi : Image Data Compression by Predictive Coding. Part II: Encoding Algorithms. IBM Systems Journal, nr 3/1974, s. 172-179 /patrz także /37/.

/8/ A.Bookstein, G.Fouty: A Mathematical Model For Estimating The Effectiveness of Bigram Coding. Information Proc. and Manag. 12, 1976 s. 111-116.

[9] C.P.Boune, D.F.Ford: A Study of Methods For Systematically Abbreviatins English Words and Names, Jour. ACM, 8, 3, July 1961 s. 53-8552.

[10] S.D.Bradley: Optimizing a Scheme of Run Lenght Encoding. Proc. IEEE /Letters/ 1969, 57, s. 108-109.

[11] H.K.Chang: Compressed Indexing Methods. IBM Technical Disclosure Bulletin II, Nr 11, 1969.

[12] T.C.Chen, I.T.Ho: Storage -Efficient Representation of Data. Comm. ACM, 18, 1, January 1975, s. 49-52.

[13] IDMS. Database Design and Definition Guide. Cullinane Corporation, September 1979, 20 William Street. Wellesley, Mass, 02181 USA.

[14] P.A.D.DeMaine: The Integral Family of Reversible Compressors Journal of the IAG, /IFIPS, Amsterdam/, 3, 1971, s.207-219.

[15] J.L.Dolby: An Algorithm Variable-Lenght Proper Name Compression. Journal of Library Automation, December 1970, s.25.

[16] S.J.Eggers, A.Shosan: Efficient Access of Compressed Data. Proceed. of Sixth Inter. Conference on VLDB, Montreal 1980, s.205-211

[17] S.J.Eggers, F.Olken, A.Shoshani: A Compression Technique for Large Statistical Databases. Proceedings of the Seventh Int. Conf. on VLDB, Cannes 1981, s. 424-434.

[18] R.Fajman, J.Borgelt:: WYLBUR: An Interactive Text Editor and Remote Job Entry System. Comm. ACM, /16/ 5, May 1973, s.314.

[19] D.W.Fokker, M.F.Lynch: Application of The Variety-Generator Approach to Searches of Personal Names in Bibliographic Data Base. Part 1. Microstructure of Personal Author's Names. Journal of Library Automation, Vol.7, nr 2/1974, s.105-118.

[20] S. W. Golomb: Run-lenght Encoding. The IEEE Trans. of Infor. Theory, IT-12, 1966, s.399.

[21] R.W.Hamming: Coding and Information Theory. Englewood Cliffs, New York, Prentice Hall 1980.

[22] A.M.Hankamer: A Modified Huffman Procedure with Reduced Memory Requirements. IEEE Trans. on Comm. June 1979, s. 930-932.

[23] J.Harker: Byte Oriented Data Compression Techniques. Computer Design, October 1982, s. 95-100.

[24] K.A.Hazboun, M.A.Bassiouni: A Multigroup Technique for Data Compression. Proc. SIGMOD 1982, s.284-292.

[25] W.D.Hagamen, D.J.Linden, H.S.Long, J.C.Weber: Verbal Information as Unique Numbers. IBM Systems Journal 11, 4 1972, s.278.

[26] B.Hahn: A New Technique For Compression and Storage of Data Comm. ACM, 17, 8, August 1974, s.434.

[27] L.J.Hoffman: Poufność w Systemach Informatycznych /Tłum z ang. WNT Warszawa 1982, s.218.

[28] D.K.Hsiao, D.S.Kerr, S.E.Madnick: Computer Security. Academic Press., New York 1979.

[29] D.A.Huffman: A.Method For The Construction of Minimum-Redundancy Codes. Proc. I.R.E. 40, 1098-1101 /1952/.

[30] R.Hunter, A.H.Robinson: International Digital Facsimile Coding Standard. Proc.IEEE, Vol. 68, 1980, s. 854-867.

[31] IBM Corporation, IBM System/370 Model 165 Functional Characteristics, Technical Newsletter, Nr GN22-0401, Juny 1971.

[32] INFORMATICS, INC., SHRINK User Reference Manual. PMI Document Nr 8616, October 1975.

[33] M.Jackson: Mnemonics. Datamation, nr 4/1967.

[34] G.C.Jewell: Text Compaction For Information Retrieval Systems IEEE SMC Newsletter, 5, 1 /Feb. 1976/.

[35] D.R.King: The Binary Vector as The Basis of an Inverted Index File. Journal of Library Automation, Vol. 7, nr 4/1974, s. 307-314.

[36] D.E.Knuth: The Art of Computer Programming vol. 3, chapt. 6.1 Addison-Wesley, 1973, s.401.

[37] H.Kobayashi, L.R.Bahl: Image Data Compression by Predictive Coding. Part I: Prediction Algorithms. IBM Systems Journal, nr 3/1974, s. 164-171 /patrz także /7/.

[38] J.Korta, J.Wojdyła: Construction and Application of Computer-aided Data Structure Analyzer. Proc. on Sixth International Seminar on DBMS. Matrafured-Hungary, 24-29.X. 1983.

[39] M.Kunt, O.Johnsen: Block Coding of Graphics: A Tutorial Review. Proc. IEEE, Vol. 68, 1980, s. 770-786.

[40] Leksykon Informatyki. Red. P. Muller, G. Lobl, H. Schmidt /Tłum. z niem./. WNT Warszawa 1977.

[41] H.Ling, F.P.Palermo: Block-oriented Information Compression. IBM Journal of Research&Development, nr 2/1975, s.141-145.

[42] C.A.Lynch, E.B.Brownrigg: Application of Data Compression Techniques to a Large Bibliographic Database. Proceedings of the Seventh Inter. Conf. on VLDB, Cannes 1981, s. 435-447.

[43] M.F.Lynch: Compression of Bibliographical Files Using an Adaptation of Run-Lenght Coding. Inf. Stor.&Retr., 9, 1972, s. 207-214.

[44] M.F.Lynch, H.J.Petrie, M.J.Snell: Analysis of The Microstructure of Titles in

The INSPEC Data-Base. Infor. Stor. Retr. Vol 9 /1973/ s. 331-337.

[45] J.Martin: Security, Accuracy and Privacy in Computer Systems. Englevood Cliffs., Prentice-Hall, 1973.

[46] J.Martin: Computer Data-Base Organization, Chapter 32, Prentice-Hall, 1975, s.713.

[47] W.R.Nugent: Compression Word Coding Techniques for Information Retrieval. Journal of Library Automation, Vol. 1, nr 4/1968, s. 250-260.

[48] M.Pechura: File Archival Techniques Using Data Compression. Comm. of ACM, nr 9/1982, s. 605-609.

[49] H.K.Reghbati: An Overview of Data Compression Techniques. Computer IEEE, Vol.14, s. 71-75 /1981/.

[50] J.J.Rissanen, G.G.Langdon: Arithmetic Coding. IBM Journal of Research and Development, vol. 23, s.149-162 /1979/.

[51] Projektowanie systemów informatycznych na komputery Jednolitego Systemu. Red. E. Niedzielska, Akademia Ekonomiczna we Wrocławiu, 1981, s. 268.

[52] T.Radhakrishnan: Selection of Prefix and Postix Word Fragment for Data Compression. Infor. Stor. Management, vol. 14,1978, s. 97-106.

[53] F.M.Reza: An Introduction To Information Theory, Chapter 4, McGraw-Hill, New York, 1961.

[54] D.G.Severance: A Practitioner's Guide to Data Base Compression . Tutorial Information Systems, nr 1/1983, s.51-63.

[55] W.S.Schieber, G.W.Thomas: An Algorithm For The Compaction of Alphanumeric Data. Journal of Library Automation 4 December 1971, s. 198-206.

[56] E.J.Schuegraf , H.S.Heaps: Selection of Equifrequent Word Fragments For Information Retrieval. Infor. Stor.&Retr. , Pergamon Press 1973, s. 697-711.

[57] E. J. Schuegraf, H. S. Heaps: A Comparison of Algorithms For Data Base Compression by The Use of Fragments as Language Elements. Infor. Stor.&Retr., 10, 1974, s. 309-319.

[58] E.J.Schuegraf: Compression of Large Inverted Files With Hyperbolic Term Distribution. Infor. Stor.&Management, Vol. 12, 1974, s. 377-384.

[59] E.J.Schuegraf, H.S.Heaps: Querry Processing in a Retrospective Document Retrieval System That Uses Word Fragments as Language Elements. Infor. Proc&Manag., vol. 12, 1976, s. 283-292.

[60] J.Seidler: Teoria Kodów. PWN Warszawa 1965, s. 310.

[61] A.Siemiński: Kompresja baz danych za pomocą kodow zmiennej długości. : Informatyka, Nr 12/1979, s. 36-39.

[62] A.J.Smith: Comments on a Paper by T.C. Chen and I.T. HO, 18, 8 August 1975, s.463.

[63] M.Snyderman, B.Hunt: The Myriad Virtues of Text Compaction. Datamation, December 1970, s. 36.

[64] D.Ting, B.Prasada: Digital Processing Techniques for Encoding of Graphics. Proc. IEEE, Vol. 68, 1980, s. 757-769.

[65] J.J.Villers, S.R.Ruth: Bibliography of Data Compaction and Data Compression Literature With Abstracts, 1971, Goverment Clearing House Study AD 723525.

[66] R.A.Wagner: Common Phrases and Minimum-Space Text Storage. Comm. ACM, 16, 3. March 1973, s. 148-152.

[67] R.E.Wagner: Index Design Consideration. IBM Systems Journal, nr 4/1973, ss. 351-367.

[68] V. R. Walker: Compaction of Names by X-grams. Proc. Am. Soc. Infor. Science 1969, 6, s.129-130.

[69] C.C.Wang: A Probabilistic Approach to Storage Compression of Large Natural Language Data Bases. Proc. COMPSAC 80. THE IEEE Conference, Chicago 1980, s. 552-555.

[70] S.F.Weiss, III R.L.Vernor: A Word-based Compression Technigue for Text Files. Journal of Library Automation, Vol.11, nr 2/1978, s. 97-105.

[71] M.Wells: File Compression Using Variable-Lenght Encodings. Computer Journal, 15, 4, November 1972, s.308.313.

[72] G.Wiederhold: Database Design. McGraw-Hill, New York, N.Y. /1977/ s. 658.

[73] J.Wojdyła: Organizacja i funkcjonowanie zbioru danych utworzonych za pomocą wirtualnej metody dostępu /VSAM/. Raport badawczy Instytutu Informatyki, II-1-82. Akademia Ekonomiczna we Wrocławiu 1982.

[74] J.Wojdyła: The Analyzer of Database Compression Methods. Proc. on Fifth Inter. Seminar On DBMS, Kołobrzeg 6-11 Dec., CPIZI Warszawa 1982, s. 12.

[75] J.Wojdyła: Kompresja danych w systemach informatycznych. Raport badawczy II-2-83, Instytut Informatyki, Akademia Ekonomiczna we Wrocławiu 1982, s. 132.

[76] J.Wojdyła: Wspomagane komputerem projektowanie procesów kompresji danych w systemach informatycznych. Prace naukowe Akademii Ekonomicznej we Wrocławiu, 1983.

mgr MIROSŁAW K.DYCZKOWSKI
Instytut Informatyki
Akademia Ekonomiczna
Wrocław

PRZEGLĄD ŚRODKÓW OCHRONY DOSTĘPU DO DANYCH W BAZACH DANYCH

Ostatnie lata przyniosły na świecie dynamiczny rozwój złożonych systemów informatycznych opartych na technologii bazy danych i związanym z nią integralnie teleprzetwarzaniu. Jedną z konsekwencji tego stanu rzeczy jest wzrost zainteresowania problemami szeroko rozumianej ochrony danych, która w warunkach systemów bazy danych, obsługujących równolegle wielu użytkowników zdalnych i lokalnych o różnych zakresach uprawnień do dostępu i manipulowania "wspólnymi" zasobami danych, nabrała nowego wymiaru.

Celem niniejszego artykułu jest dokonanie przeglądu i ogólnej charakterystyki środków ochrony dostępu do danych stosowanych w systemach bazy danych. Ze względu na ograniczone rozmiary opracowania omawiane są przede wszystkim środki ochrony charakteryzujące się najwyższą skutecznością działania, które są zaimplementowane w większości komercyjnych sieciowych systemów zarządzania bazą danych /SZBD/.

Podstawowe pojęcia i definicje

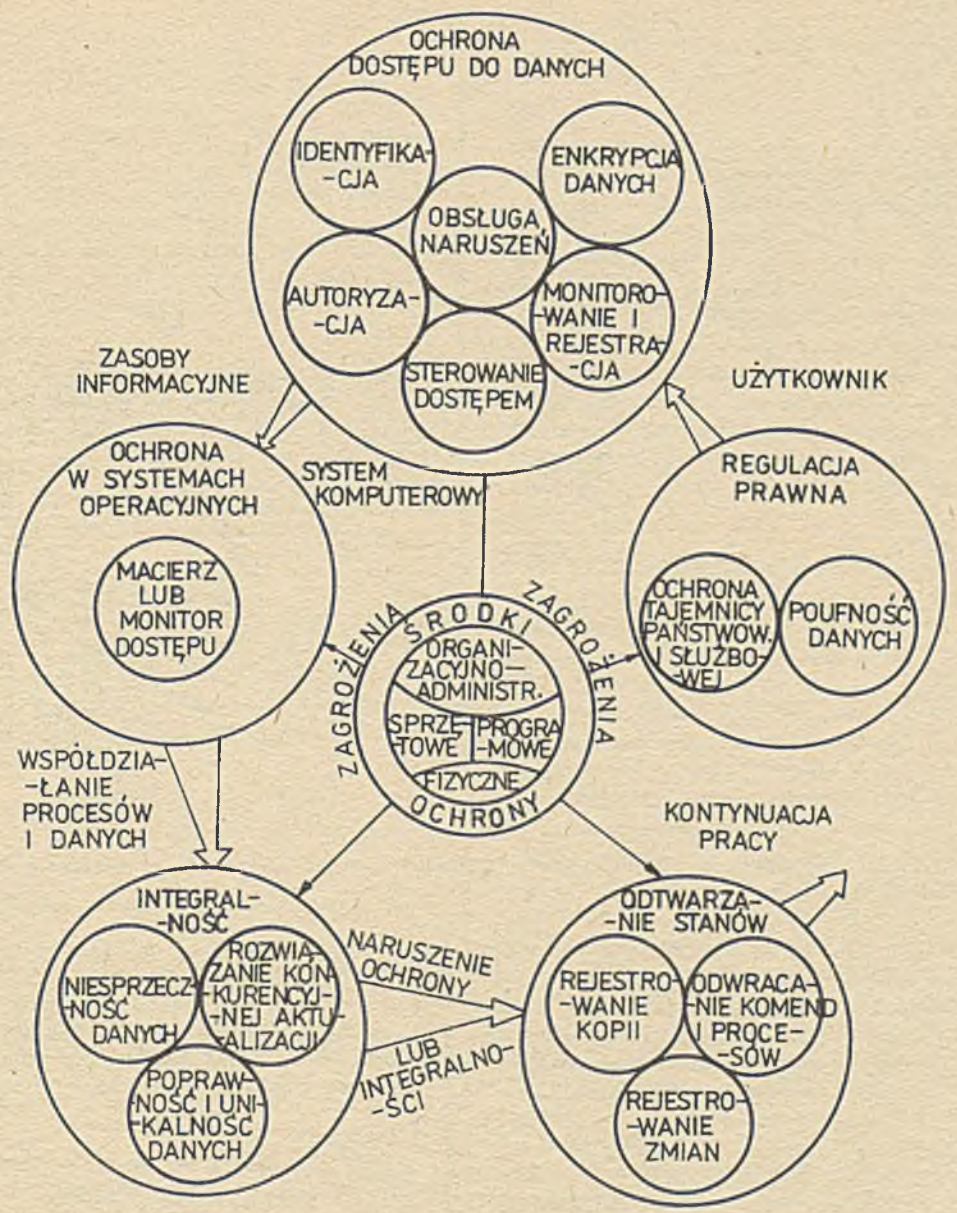
W związku z tym, iż brak jest w dziedzinie ochrony danych standardów terminologicznych w celu jednoznacznego rozumienia używanych w artykule pojęć podamy na wstępie kilka definicji. Dla potrzeb artykułu można przyjąć, że ochrona danych, to system współzależnych działań dokonywanych na pełnym zbiorze dostępnych środków zapewniających bezpieczeństwo i nienaruszalność danych przetwarzanych lub przechowywanych w systemie bazy danych i umożliwiających właściwy dostęp do tych danych. Przez środki ochrony danych rozumiemy rozwiązania sprzętowe, programowe, fizyczne i organizacyjno-administracyjne, które można zastosować do sprzętu, programów i danych, w celu osiągnięcia ich bezpieczeństwa i nienaruszalności /3/.

Tak rozumiana ochrona danych obejmuje zagadnienia regulacji prawnej w dziedzinie ochrony informacji przetwarzanej w systemach bazy danych oraz takie problemy technologiczne jak: ochrona dostępu do danych, ochrona w systemach i podsystemach operacyjnych, integralność danych i bazy danych a także odtwarzanie stanów bazy danych /rys.1/ /5,12,16,20, 24/. Przedmiot dalszych rozważań stanowi wyłącznie ochrona dostępu do danych. Jest to zespół działań polegających na definiowaniu kiedy, jak i w jakim zakresie przechowywane lub przetwarzane w systemie dane mogą być dostępne dla określonych użytkowników oraz zabezpieczenia tych danych przed nieupoważnionym dostępem przez izolację danych, sterowanie i kontrolę dostępu oraz manipulowania danymi, a także ich enkrypcję. Praktyczna realizacja tak rozumianej ochrony dostępu do danych sprowadza się do włączenia do systemu zarządzania bazą danych i środków z nim bezpośrednio współdziałających oraz go wspomagających, mechanizmów programowych lub możliwości organizacyjnych zapewniających obsługę następujących funkcji:
- identyfikacji i autoryzacji,
- sterowania dostępem,
- monitorowania i rejestracji,
- enkrypcji,
- obsługi naruszeń /8/.

Kompletny, spójny i wewnętrznie niesprzeczny zbiór zabezpieczeń tworzy system ochrony dostępu do danych. Ogólny schemat działania takiego systemu w bazie danych ilustruje rys.2.

Identyfikacja i autoryzacja

Celem identyfikacji na etapie projektowania systemu bazy danych jest określenie wszystkich podmiotów i przedmiotów ochrony oraz ich atrybutów, a także włączenie do systemu zabezpieczeń zestawu środków obsługujących procedury identyfikacji. Na etapie eksploatacji identyfikacja obejmuje ciągłe lub wyrywkowe pobieranie i porównywanie identyfikatorów podmiotów i przedmiotów ochrony, które uczestniczą w operacjach dostępu. Identyfikatory są to niepowtarzalne łańcuchy znakowe lub bitowe, które są przypisywane wszystkim użytkownikom, urządzeniom /zwłaszcza terminalom/, programom, elementom struktur danych i opisów tych struktur występującym w środowisku bazy danych. Są one pobierane przez procedury identyfikacyjne realizowane w czasie inicjacji sesji oraz przy wszystkich próbach dostępu i manipulowania elementami struktur bazy danych, jej opisów lub programami przechowywanymi w bibliotekach systemu bazy danych. Identyfikatory są używane do sterowania dostępem oraz

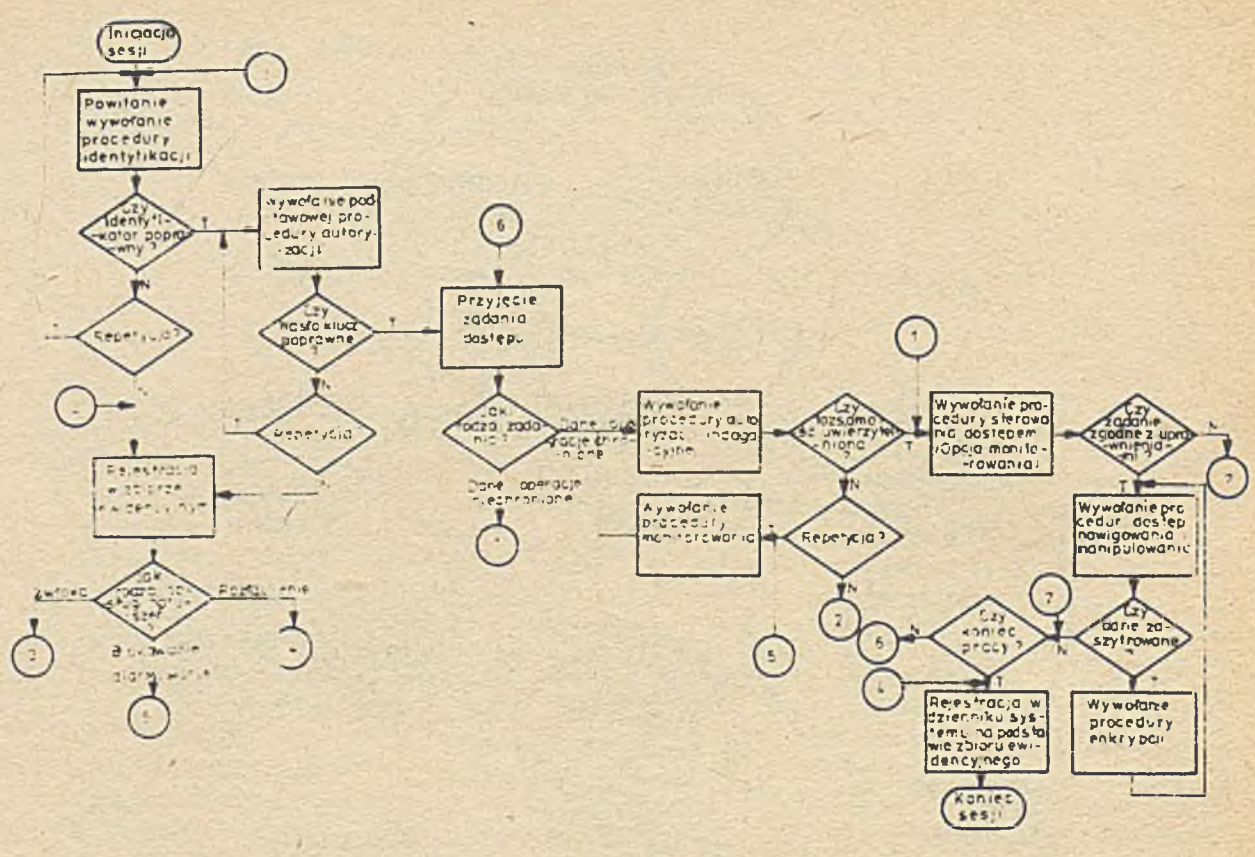


Rys. 1. Podstawowe aspekty ochrony danych /Źródło: opracowanie własne/

monitorowania i rejestracji zdarzeń i stanów występujących w czasie działania systemu [19].

W związku z tym, iż identyfikatory podają wyłącznie niepotwierdzoną tożsamość muszą być one w przypadku, gdy w systemie bazy danych jest pożądany jakikolwiek stopień bezpieczeństwa, dodatkowo autoryzowane. Autoryzacja na etapie projektowania systemu polega na zdefiniowaniu atrybutów używanych do potwierdzania tożsamości obiektów ochrony oraz włączeniu zestawu środków obsługujących procedury autoryzacji. Celem autoryzacji w czasie eksploatacji systemu bazy danych jest jednorazowe, okresowe, wyrywkowe lub stałe potwierdzanie tożsamości wszystkich zidentyfikowanych podmiotów i przedmiotów ochrony [27].

Realizacja identyfikacji i autoryzacji w przypadku pracy wsadowej jest zapewniona przez wbudowane w Język Opisu Danych /JOD/ schematu i podschematu mechanizmu blokad dostępu /klauzule PRIVACY LOCK/ i współdziałające z nimi, wchodzące w skład Języka Manipulacji Danymi /JMD/ klucze dostępu /komenda PRIVACY KEY/. Ze względu na fakt, iż mechanizmy blokad i kluczy dostępu tylko w



Rys. 2. Ogólny schemat działania systemu ochrony dostępu do danych w systemie bazy danych. /Źródło: opracowanie własne/

ograniczonym stopniu są zorientowane na identyfikację i autoryzację zadań użytkowych, a ich podstawowym zadaniem jest zapewnienie na poziomie schematu i podschematu oraz wielozadaniowym monitorze bazy danych sterowania dostępem, szczegółowo zostaną one omówione przy opisie drugiej grupy funkcji. Znacznie szerzej funkcje identyfikacji i autoryzacji użytkownika są spełniane w przypadku pracy bezpośredniej, także w środowisku telekomunikacyjnym, gdzie odpowiednie zabezpieczenia są wbudowane w monitory telekomunikacyjne SZBD, procesory języków zapytań lub inne oprogramowanie umożliwiające zdalny i lokalny dostęp i manipulowanie danymi.

Zabezpieczenia te umożliwiają opcjonalnie:
- identyfikację operatora /grupy operatorów/ terminala, poprzez przydzielony mu identyfikator,
- autoryzację dostępu operatora /grupy operatorów/ terminala, poprzez hasło dostępu i dialog indagacyjny.
- określenie elementów struktur danych, które mogą być udostępnione użytkownikowi korzystającemu z języka zapytań /13,15/.

Szerszego omówienia wymagają ze względu na rozpowszechnienie hasła: dostępu i dialog indagacyjny. H a s ł o d o s t ę p u to łańcuch znakowy wprowadzany przez użytkownika i porównywany przez procedurę autoryzacji z wzorcem przechowywanym w chronionym zbiorze systemowym. Hasła mogą mieć charakter pasywny i aktywny. Hasła pasywne są wprowadzane przez użytkownika jako pełne łańcuchy znakowe zgodne z wzorcem systemowym /26/.

Przykład 1
logon, mkd Ø83
ENTER PASSWORD: 6Ø653
OK

Możliwe jest też stosowanie pewnych modyfikacji haseł pasywnych polegających na wycinaniu wybranych znaków czy stosowaniu haseł jednorazowych.

Przykład 2
logon, mkd Ø83
ENTER PASSWORD, PODAJ ZNAKI 1 i 4 : 65
OK

Użytkownik podaje w tym przypadku wyłącznie określone znaki z hasła, których numery są losowo generowane przez procedurę autoryzacji, na przykład na podstawie przekształcenia stanu zegara. Natomiast hasła aktywne są to łańcuchy znakowe wyprowadzane przez system, na których użytkownik musi dokonać żądanych operacji arytmetycznych lub logicznych i ich wynik przesłać do systemu w celu autoryzacji tożsamości /26/.

Przykład 3
logon, mkd Ø83
ENTER PASSWORD, 173 : 116
OK

Użytkownik w tym przypadku dokonuje transformacji łańcucha n_1, n_2, n_3 wyprowadzonego przez system zgodnie ze znanym sobie algorytmem: dodaj $n_1+n_2+n_3$ i dołącz $n_{max} - n_{min}$ i podaje jej wynik. Hasła aktywne są bardziej skomplikowane dla użytkownika i czasochłonne od pasywnych. Gwarantują jednak wyższy poziom bezpieczeństwa.

Omawiając hasła dostępu należy zwrócić uwagę na trzy problemy:
1. Hasła nigdy nie powinny być przechowywane w zbiorach systemowych, kolejkach itp. w postaci jawnej, muszą być zaszyfrowane i to przy pomocy szyfrów nieodwracalnych /14/.

2. Hasła powinny być odpowiednio często zmieniane, przy czym bezpieczny czas posługiwania się hasłem jest tym krótszy im krótsze jest hasło i im mniejsza jest liczebność alfabetu, z którego jest ono tworzone. Obliczając ten czas można posługiwać się znanym wzorem Andersona:

$$4{,}32 \cdot 10^4 \frac{R*M}{E*P} < A^S$$

gdzie:
R jest szybkością przesyłania w znakach na minutę,
E jest liczbą znaków wymienianych przy każdorazowej próbie włączenia się do systemu,
S jest długością hasła w znakach,
A jest liczebnością alfabetu, z którego jest tworzone hasło,
P jest pożądanym prawdopodobieństwem złamania hasła,
M jest czasem dokonywania prób złamania w miesiącach całodobowych /1/.

3. Hasła powinny być maskowane przez stosowanie pól niewyświetlanych w przypadku terminali typu monitor ekranowy lub nadpisanie na drukarkach /21/.

Wyższy poziom bezpieczeństwa od haseł zapewnia autoryzacja poprzez d i a l o g i n d a g a c y j n y. Polega ona na zadawaniu użytkownikowi przez system pytań generowanych z listy przechowywanej w systemie, na które musi on odpowiedzieć poprawnie w celu uwierzytelnienia tożsamości. Pytania powinny być tak dobrane, aby odpowiedzi na nie były znane wyłącznie określonemu użytkownikowi. /34/.

Przykład 4
logon, mkd Ø83
PODAJ NUMER REJESTRACYJNY SAMOCHODU: wid 7494
NA KTORYM PIETRZE MIESZKASZ: 8
IMIE ŻONY: Elżbieta
OK

W systemach o dużej liczbie użytkowników jest to metoda czaso i pamięciochłonna. Z tego względu jest ona stosowana jako drugi, dodatkowy poziom autoryzacji w przypadku pracy z danymi lub programami sensytywnymi. Może też być zmodyfikowana przez wprowadzenie standardowej listy pytań systemowych, z których użytkownicy wybierają te, które ich zdaniem najlepiej zabezpieczają proces autoryzacji. Przechowywanie w tym przypadku wartości odpowiedzi wraz z łącznikami do wybranych pytań eliminuje potrzebę pamiętania pytań każdego użytkownika, przyczyniając się do oszczędności pamięci /36/.

Sterowanie dostępem

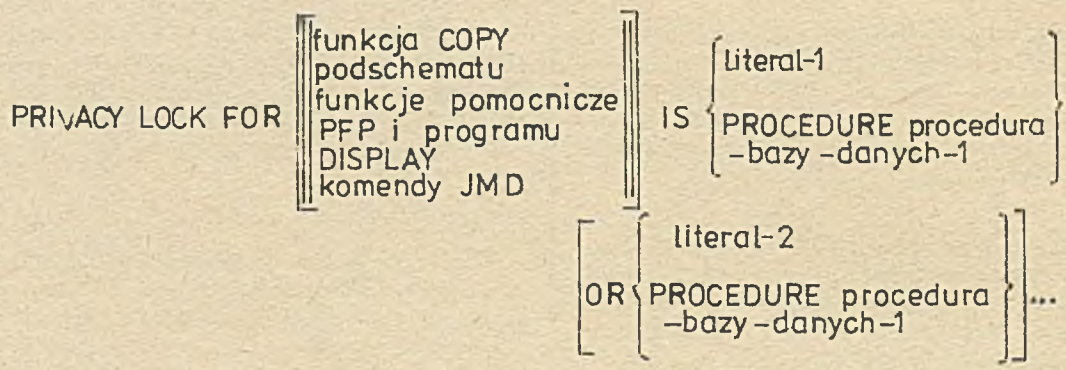
Po dokonaniu autoryzacji obiektów ochrony należy zweryfikować ich uprawnienia w zakresie dostępu do określonych elementów struktur danych lub programów i dopiero na tej podstawie wykonać żądane operacje. Zagadnienie to jest przedmiotem sterowania dostępem. Sterowanie dostępem czyli upoważnianie polega na etapie projektowania systemu bazy danych na zdefiniowaniu warunkowych i bezwarunkowych praw dostępu podmiotów do przedmiotów ochrony przez przyporządkowanie każdej relacji użytkownik - terminal - program - dane określonego typu dostępu oraz włączenie zestawu środków realizujących te prawa. W czasie eksploatacji systemu prowadzona jest ciągła, bieżąca kontrola upoważnień, a dostęp jest przyznawany wyłącznie zgodnie ze zdefiniowanymi prawami /22/.

Jak poprzednio wspomniano w przypadku pracy wsadowej realizacja tych funkcji jest zapewniona przez wbudowanie w JOD schematu i podschematu mechanizmu blokad dostępu i współdziałające z nimi wchodzące w skład podschematu, funkcje pomocnicze oraz JMD klucze dostępu. Klucz dostępu jest to pojedyncza wartość o zdefiniowanej przez implementora wielkości lub typie, która może być stałą wartością zmiennej albo wynikiem działania procedury bazy danych.

Blokada dostępu jest wartością /stałą lub zmienną/ zdefiniowaną podczas opisu struktury logicznej bazy danych, porównywaną z dostarczoną przez zadanie użytkowe wartością klucza dostępu lub procedurą, która jest przywoływana i ma dostęp do właściwego klucza. Powrót procedury z wynikiem pozytywnym umożliwia dostęp do chronionych elementów danych. Oprócz wymienionej funkcji sterowania dostępem procedura może realizować:
- dodatkowe obliczenia na kluczu w celu sprawdzenia jego ważności,
- inicjację dialogu indagacyjnego z użytkowni-

kiem terminala w celu dodatkowej autoryzacji,
- wpisanie nowej wartości klucza do zadania,
- obsługę naruszeń dostępu przez zawieszenie lub przerwanie zadania, rozłączenie terminala, alarmowanie konsoli administratora ochrony itp.

Blokady dostępu są deklarowane w klauzulach typu PRIVACY LOCK, które mają następujący format ogólny:

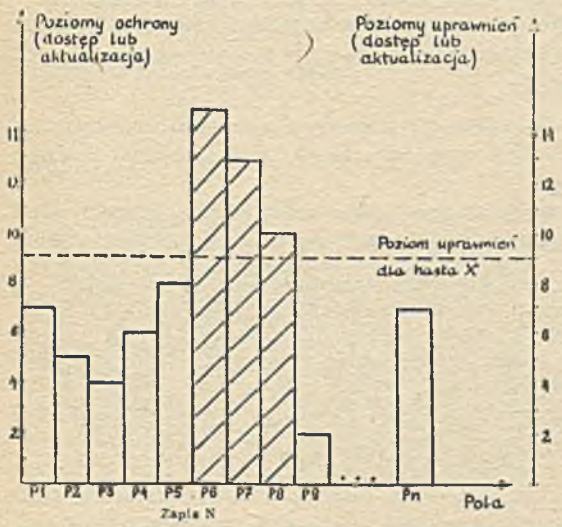


Mogą być one ustanawiane dla dwóch różnych grup rodzajów dostępu:
1. dla sterowania dostępem do opisów struktury logicznej i fizycznej bazy danych na poziomach:
- schematu - zakres ich działania dotyczy operacji kopiowania części lub całości schematu /COPY/ wprowadzania zmian do schematu /ALTER/, drukowania opisu bazy danych /DISPLAY i zmiany blokad /LOCKS/,
- podschematu - zakres ich działania dotyczy kompilacji programów przywołujących dany podschemat /COMPILE/, wprowadzania zmian w podschemacie /ALTER/, drukowania opisu części bazy ograniczonej podschematem /DISPLAY/ oraz zmiany blokad /LOCKS/,
2. dla sterowania dostępem do elementów struktury logicznej i fizycznej bazy danych na poziomach:
- obszaru - zakres ich działania wyłącznego lub chronionego otwarcie obszaru dla operacji wyszukiwania lub aktualizacji lub dla każdej z funkcji PFP /INTIATE, PRINT, PATCH, VERIFY/,
- zapisu - zakres ich działania dotyczy komend JMD operujących na zdefiniowanym typie zapisu /komendy: INSERT, REMOVE, STORE, DELETE, GET, MODIFY, FIND/,
- zbioru bazy danych - zakres ich działania dotyczy komend JMD operujących na zdefiniowanym typie zbioru /komendy: FIND, INSERT, REMOVE/,
- zapisu podrzędnego - zakres ich działania dotyczy komend JMD operujących na wystąpieniach zapisów podrzędnych zdefiniowanego typu zbioru /komendy: FIND, INSERT, REMOVE/,
- danej elementarnej i złożonej - zakres ich działania dotyczy komend JMD operujących na zdefiniowanych danych /komendy: STORE, GET, MODIFY/,
- wystąpień danej elementarnej lub złożonej określanych przez zakres wartości - wprowadza się w tym celu do klauzuli PRIVACY dodatkową opcję WHERE /8,28,30,31/.

Blokady dostępu ustanowione w schemacie bazy danych dla wymienionych poziomów i zakresów ochrony mogą być zmienione w podschematach. Możliwe jest to dla poziomów i obszaru, zapisu, zbioru, zapisu podrzędnego i danej. Pozwala to na deklarowanie odrębnych zbiorów praw dostępu dla zadań użytkowych posługujących się różnymi podschematami. Rozszerza się w ten sposób elastyczność systemu ochrony dostępu /18/.

W przypadku pracy bezpośredniej w środowisku lokalnym lub zdalnym SZBD używają do sterowania dostępem opisanych wyżej mechanizmów blokad i kluczy dostępu. W niektórych SZBD korzysta się także z metody poziomów uprawnień /9/. Istotą metody poziomów uprawnień jest przydzielenie, na podstawie przeprowadzonej wcześniej analizy struktur bazy danych każdemu najmniejszemu chronionemu elementowi danych /pole, dana, elementarna, złożona, zapis/ określonych wielkości numerycznych pełniących funkcję poziomów ochrony /security levels/.
Wielkości te wybierane są z przedziału o ustalonej skali rozpiętości /na przykład 0-15/. W najprostszej wersji przydziela się każdemu elementowi danych po dwa poziomy ochrony dla dostępu /ACCESS/ i aktualizacji /UPDATE/. W bardziej skomplikowanych systemach możliwe jest ustalenie poziomów ochrony dla wybranych lub wszystkich komend manipulowania danymi, takich jak: INSERT, REMOVE, STORE, DELETE, GET, MODIFY, FIND. Następnie,

na podstawie analizy potrzeb informacyjnych poszczególnych użytkowników przydziela się im hasła dla współpracy z określonymi elementami struktur danych lub realizacji procedur. Do każdego hasła zostają przyporządkowane wielkości numeryczne pełniące funkcje poziomów uprawnień /permission levels/. Wielkości te wybierane są podobnie jak w przypadku poziomów ochrony z przedziału o ustalonej skali rozpiętości, która jest jednak o jeden mniejsza od skali dla poziomów ochrony /w naszym przypadku wynosi 0-14/. Poziomy uprawnień również w minimalnym zakresie specyfikuje się dla dostępu i aktualizacji, w innych przypadkach dla tych wszystkich komend manipulowania danymi, dla których określono poziomy ochrony. Żaden z chronionych elementów danych nie zostanie udostępniony zadaniu użytkowemu, które posługuje się hasłem posiadającym poziom uprawnień niższy od poziomu ochrony. Dodatkowy, jeden rezerwowy poziom ochrony /najwyższy/ wykorzystywany jest do uniemożliwienia dostępu lub manipulowania danymi, które system wykorzystuje dla celów ochrony danych /systemowe tablice haseł, procedury identyfikacji, kontroli uprawnień, obsługi naruszeń/. Schemat działania metody poziomów uprawnień przedstawiony jest na rys.3.



Rys. 3. Schemat działania metody poziomów uprawnień dla zapisu N i hasła X /Źródło: opracowanie własne

Dla poszczególnych pól /P1, P2,..., Pn/ zapisu N przydzielono poziomy ochrony dla dostępu lub aktualizacji /SL. P1-7, SL. P2=5,..., SL.Pn=7/. Użytkownik /użytkownicy/ posługujący się przy dostępie do zapisu N lub przy jego aktualizacji hasłem X wykorzystać będą mogli tylko te pola danych, które posiadają niższy poziom ochrony od poziomu uprawnień hasła X, który wynosi 9. Nie uzyskają więc dostępu do pól P6, P7 i P8 /na rysunku zaciemnione/. Użytkownik, który posługiwałby się hasłem o poziomie uprawnień 1, nie uzyska dostępu do żadnego z pól danych, natomiast żaden z użytkowników nie może wykorzystać pola P6, gdyż jego poziom ochrony wynosi 15 i jest większy od najwyższej wielkości poziomu uprawnień. Każde hasło np. X może posiadać inne uprawnienia dla różnych elementów struktur danych, procedur lub komend manipulowania danymi.

Zarówno blokady oraz klucze dostępu, jak i metoda poziomów uprawnień umożliwiają wyłącznie definiowanie i implementowanie bezwarunkowych praw dostępu i manipulowania danymi. Oparte są one bowiem na prostej macierzy dostępu typu ACB /access code byte/, w której wszystkie dozwolone operacje dostępu i manipulowania są odwzorowane za pomocą łańcuchów bitowych lub znakowych, ewentualnie haseł. W wielu systemach bazy danych takie proste macierze dostępu są niewystarczające /25/, warunkowych praw dostępu. Warunkowe prawa dostępu, które uzależniają upoważnienie oraz jego typ od czynników nie będących wyłącznie elementami relacji między podmiotem a przedmiotem ochrony, pozwalają na znaczne uelastycznienie sterowania dostępem. Zdefiniowanie warunkowych praw dostępu pozwala na uwzględnienie przy upoważnieniu takich czynników jak:

- żądana wartość wystąpień zasobów, zwłaszcza danych,
- dynamiczny stan systemu,
- historia dostępów użytkownika do danego lub innych zasobów
- zalecany sposób użytkowania zasobów,
- wartość dodatkowych zmiennych systemowych, zwłaszcza takich jak czas, częstotliwość czy lokalizacja dostępu /10,11/.

Aby to osiągnąć tworzy się złożone macierze dostępu, które zawierają obok łańcuchów bitowych i znakowych także odsyłacze do procedur.

W związku z tym, że macierze dostępu są w systemach baz danych pamięciochłonne próbuje się je redukować wprowadzając macierze zredukowane. Redukcja ta polega na stosowaniu takich metod jak: definiowanie grup "wirtualnych" użytkowników o identycznych upoważnieniach, definiowanie grup "wirtualnych" terminali, transakcji lub programów o tym samym poziomie uprawnień, grupowanie danych w określone kategorie ochrony, przechowywanie zbioru par. użytkownik /terminal lub program/ - pozwolenie dla każdego elementu danej oraz zbioru inwersyjnego: element danej - pozwolenie, dla każdego użytkownika /terminala, programu/ /35/.

Dodatkowymi środkami ochrony wspomagającymi przedstawione wyżej mechanizmy ste-

rowania dostępem SZBD są takie działania organizacyjno-administracyjne jak:
- separacja danych, polegająca na umieszczaniu w oddzielnych typach elementów struktury danych wystąpień danych sensytywnych,
- izolacja danych identyfikacyjnych od wystąpień danych sensytywnych, na przykład przez zorganizowanie ich w różnych typach zbiorów z zastosowaniem dodatkowego zbioru pośredniczącego, składającego się wyłącznie z łączników adresowych lub wtórnych identyfikatorów zapisów z danymi,
- stosowanie obowiązkowego pośrednictwa, czyli wprowadzenie środków ochrony dla wszystkich ścieżek selekcji danych /3,4,17/.

Monitorowanie i rejestracja

Monitorowanie i rejestracja zdarzeń i stanów występujących w czasie realizacji operacji dostępu ma na celu bieżącą kontrolę działania procedur zabezpieczających oraz gromadzenia danych umożliwiających wykrycie słabych miejsc w systemie ochrony. Na etapie projektowania systemu bazy danych ta grupa funkcji obejmuje określenie listy akcji systemowych, które mają być monitorowane i rejestrowane, ustalanie zakresu zbieranych danych oraz włączenie do systemu zestawu środków śledzących i rejestrujących. W czasie eksploatacji systemu są natomiast prowadzone: ciągła, bieżąca rejestracja zdarzeń i stanów w dzienniku ochrony oraz wyrywkowe monitorowanie operacji dostępu i manipulowania danymi /23, 24/.

Należy podkreślić, że bieżące śledzenie zdarzeń i stanów zachodzących w trakcie procesu obsługi żądań użytkowników jest jedną z najważniejszych i najbardziej skutecznych metod podnoszenia poziomu bezpieczeństwa w systemie bazy danych. Pozwala ono bowiem administratorowi ochrony, kontrolującemu aktualnie pracę systemu bazy danych przy pomocy specjalnego terminala /typu monitor ekranowy lub drukarka z klawiaturą/, na podejmowanie określonych działań w sytuacjach, gdy standardowa, automatyczna obsługa naruszeń nie jest wystarczająca. W większości SZBD są opcje przesyłania do terminala ochrony komunikatów dotyczących wszystkich akcji związanych z identyfikowaniem i autoryzowaniem obiektów, upoważnianiem oraz wywoływaniem procedur specjalnych /np. enkrypcji danych/, zwłaszcza przy niepomyślnych próbach wykonania lub przy działaniu na danych sensytywnych. Samo informowanie o stanie i zdarzeniach rozwiązuje jednak problem połowicznie. Musi bowiem istnieć zespół środków programowo-sprzętowych i organizacyjnych, pozwalających administratorowi nadążnie oddziaływać na śledzony proces. Stosowane w tym celu w SZBD mechanizmy są przedstawione w dalszej części artykułu, przy opisie obsługi naruszeń.

Opisane monitorowanie jest łączone z rejestracją danych dotyczących wymienionych rodzajów akcji, a więc z prowadzeniem dziennika systemu. Bez dokładnego dziennika jest niemożliwe zbadanie tego, co wydarzyło się w przeszłości i usuwanie tych błędów w funkcjonowaniu zabezpieczeń, które się już ujawniły. Prowadzenie dokładnych dzienników na poziomie systemu operacyjnego, SZBD lub systemu ochrony dostępu jest względnie proste, gdyż współczesne SZBD pozwalają na rejestrację wszystkich nieskutecznych i skutecznych prób uzyskania dostępu do danych i programów z ich pełną charakterystyką. W związku z tym, że zakres i tryb rejestracji zapisów w dzienniku jest bardzo często pozostawiony do decyzji projektanta konkretnego systemu bazy danych należy podkreślić, że dane zawarte w dzienniku powinny pozwolić na to, aby dla dowolnego przedziału czasu można było ustalić:

- pełne opisy upoważnień związanych z każdym chronionym zasobem,
- odpowiedzialnych za zmiany dokonane w tych opisach,
- zakres i wartość tych zmian,
- wszystkie przypadki dostępu do chronionych zasobów uporządkowane według identyfikatorów użytkownika, programu, urządzenia, zadania, danych, typu dostępu ewentualnie innych żądanych kluczy /np. czasu, częstości i lokalizacji dostępu/,
- wszystkie odmowy udzielania dostępu uporządkowane według podanych wyżej identyfikatorów, przy czym dodatkowo można badać punkt bezpieczeństwa, w którym nastąpiła odmowa,
- wszystkie tymczasowe upoważnienia wykraczające poza opis zawarty w macierzy dostępu,
- wszystkie przestawienia zegara i inne zmiany pamięci dokonane przez operatora /19,27/.

Przykładową strukturę zapisu dziennika spełniającego powyższe zadania zawiera tabela 1. Jak z niej wynika, możliwe jest wykorzystanie do celów utrzymywania bezpieczeństwa typowych dzienników systemowych lub rozliczeń należności z tym, że trzeba wyszczególnić dodatkowe typy zapisów.

Na podstawie omówienia zawartości dziennika systemu można zauważyć, że może być on stosowany także do innych celów niż wspomaganie eksploatacji bieżącej systemu ochrony dostępu do danych. Dostarcza on bowiem danych do strojenia i optymalizacji systemu i jego procedur, a przede wszystkim w przypadku rejestrowania stanu zasobów przed i po zmianach /aktualizacji, modyfikacjach, kasowaniach itp./ - jest narzędziem odtwarzania bazy danych. Poza tym dziennik jest stosowany do wspierania innych środków ochrony dostępu. Pomaga m.in.: śledzić proces starzenia się haseł i ostrzega o kończeniu się bezpiecznego

czasu ich używania. Dostarcza także danych o wykorzystaniu określonych typów zasobów, które służą do modyfikacji macierzy dostępu. Trzeba również podkreślić psychologiczne oddziaływanie funkcjonowania procedur monitorowania i rejestracji na potencjalnych intruzów, jako czynnika odstraszającego /2/.

Tabela 1

Struktura zapisu w dzienniku ochrony SZBD

Typ zapisu	Czas	Urządzenie	Użytkownik	Zadanie, krok program	Zdarzenie
93	10:23:51	TØØ16	UØ312 Kowal Jan	PØ1 Ø7	niepoprawny klucz dostępu do otwarcia obszaru PRACOWNIK, wartość: APRA 437

Źródło: opracowanie własne.

Enkrypcja

Enkrypcja czyli szyfrowanie danych jest sposobem ochrony polegającym na przechowywaniu danych lub programów w postaci utrudniającej maksymalnie ich wykorzystanie po uzyskaniu do nich nieupoważnionego dostępu. Zadaniem enkrypcji na etapie projektowania systemu bazy danych jest określenie i włączenie do systemu sprzętowych lub programowych mechanizmów szyfrujących wybrane elementy struktur bazy danych. W czasie eksploatacji systemu enkrypcja polega na kodowaniu i dekodowaniu elementów danych oraz stałej weryfikacji działania mechanizmów szyfrujących /7/. Realizację tej funkcji ochrony dostępu w sieciowych SZBD, w przypadku pracy wsadowej i bezpośredniej, zapewnia specyfikowana w schemacie w zadaniu RECORD specjalna procedura konwersji. Definiowana jest ona w klauzuli ENCODING/DECODING, która ma postać:

FOR { ENCODING / DECODING } [ALWAYS] CAL procedura-bazy-danych-1

Specjalna procedura konwersji przy określeniu opcji ENCODING jest wywołana podczas wykonywania komend JMD:STORE - zapisanie do bazy danych nowego wystąpienia zapisu danego typu i MODIFY - modyfikacja wystąpienia zapisu danego typu. Natomiast, gdy zostanie podana opcja DECODING wywołanie procedury konwersji jest realizowane podczas wykonania komendy GET przesyłającej wystąpienie z bazy do obszaru roboczego użytkownika. Tak funkcjonująca klauzula ENCODING/DECODING wykonuje w ograniczonym zakresie enkrypcję danych, a tym samym wzmacnia system zabezpieczeń. Widać więc, że skuteczność szyfrowania zależy od tego na ile złożone algorytmy szyfrujące zostaną opisane i oprogramowane przez użytkownika w wywołanej przez klauzulę ENCODING/DECODING procedurze bazy danych. Istnieje również możliwość zastosowania jako quasi enkrypcji metod kompresji baz danych. W związku jednak z tym, iż w większości systemów bazy danych realizowanych dla przechowywania danych gospodarczych enkrypcja ma ze względu na znaczną czaso i pamięciochłonność oraz subiektywnie niską ocenę cenność informacji ograniczone zastosowanie, odsyłamy zainteresowanych metodami enkrypcji do opracowań szczegółowych /7,19,20,21,24,36/.

Obsługa naruszeń

Obsługa naruszeń ochrony dostępu do danych obejmuje wyznaczenie listy zdarzeń, które będą wywoływać aktywne i pasywne akcje zapobiegawcze oraz włączenie do systemu bazy danych odpowiednich transakcji i procedur. W większości SZBD obsługa naruszeń polega na:

- zapewnieniu możliwości repetycji /najczęściej 2 lub 3-krotnej/ w przypadku wystąpienia błędów w trakcie realizacji identyfikacji czy autoryzacji, z odpowiednią sygnalizacją błędu użytkownikowi i obsłudze systemu,
- blokowaniu lub rozłączaniu urządzeń, z których realizowane są próby uzyskania nieupoważnionego dostępu,
- zwłoce w wykonywaniu operacji dostępu niezgodnej z uprawnieniami, w czasie której nowe próby uzyskania dostępu są ignorowane oraz alarmowany jest administrator ochrony,
- zawieszaniu lub kasowaniu programów lub transakcji naruszających ustalone prawa dostępu,
- wbudowaniu do oprogramowania systemu specjalnych transakcji lub procedur umożliwiających administratorowi bieżące i interakcyjne zamykanie lub zmianę parametrów procesora języka zapytań, monitora telekomunikacyjnego

Tabela 2

Wartość selektywności i wnikliwości ochrony w wybranych Systemach Zarządzania Bazą Danych

Składniki parametrów ochrony / SZBD	Minimalny poziom ochrony			Minimalny zakres ochrony				Minimalne kryterium dopuszczalności		Uwagi
	baza danych	grupa danych	dana	pełna strukt. danych	typ elementu struktury: grupa danych	dana	wartość elementu struktury	proces	typ operacji	
ADABAS			dana elementarna				wystąpienie danej elementarnej		komenda JMD	Opcja szyfrowania wystąpień danych elementarnych
System 2000			składnik			typ składnika			komenda JMD	
IMS/VS		segment			typ segmentu				komenda DL/I	
TOTAL	baza danych			baza danych				program		ochrona przez system operacyjny
ROBOT		obiekt danych			typ obiektu			zadanie		
DMS 1100	baza danych			baza danych				program		ochrona przez system operacyjny
IDMS		zapis			typ zapisu				komenda JMD	
RODAN			dana elementarna			typ danej elementarnej			komenda JMD	klauzula konwersji danych

Źródło: Opracowanie własne

lub innego oprogramowania sterującego dostępem bezpośrednim, uniemożliwianie wykonywania transakcji z kolejek systemowych, wyświetlanie informacji o transakcji bieżącej powodowanie anormalnego zakończenia transakcji bieżącej oraz interakcyjną zmianę haseł i poziomów uprawnień /6,32,33/.

Przedstawiony przegląd środków ochrony dostępu do danych objął przede wszystkim te, które są implementowane w większości komercyjnych SZBD, w tym zwłaszcza w systemach opartych na modelu sieciowym. Kończąc opis spróbujemy dokonać syntetycznego porównania możliwości dostarczonych w zakresie ochrony dostępu przez wybrane komercyjne SZBD. Porównanie jest oparte na opracowanych przez autora strukturalnych parametrach ochrony: selektywności i wnikliwości.

Selektywność ochrony określa na jakich poziomach i w jakich zakresach funkcjonują poszczególne środki ochrony dostępu do danych, a tym samym determinuje możliwości ich używania dla poszczególnych typów elementów struktury danych systemu bazy danych. Szczególnie istotne przy opisie selektywności ochrony jest określenie najmniejszych elementów danych /usytuowanych najniżej w hierarchii danych/, dla których można stosować zabezpieczenia o minimalnym zakresie działania, czyli wyznaczenie tzw. granicznej selektywności ochrony. Wnikliwość ochrony służy do określenia typów działań,które mogą być objęte przez określony środek ochrony lub cały system zabezpieczeń wspólnym kryterium dopuszczalności wykonania, wyznaczonym dla określonego poziomu i zakresu ochrony, przy czym szczególnie ważne jest wyznaczenie wartości tego parametru dla granicznej selektywności ochrony. Chodzi tu zwłaszcza o możliwości stosowania minimalnych kryteriów dopuszczalności dla poszczególnych wystąpień elementów danych opisanych przez wartość, identyfikator lub też przedział wartości.

Selektywność i wnikliwość ochrony łącznie determinują więc możliwości implementacyjne sterowania dostępem, wyznaczając rodzaje typów dostępu w danym systemie zabezpieczeń. Przykładowe wartości tych parametrów zapewniane we współczesnych SZBD zawiera tabela 2, opracowana na podstawie wcześniejszych analiz autora /8/. Wynika z niej, że wartość parametrów strukturalnych jest w poszczególnych SZBD zróżnicowana. Minimalny poziom ochrony mieści się w przedziale dana elementarna /ADABAS, System 2000, RODAN/ - baza danych /TOTAL, DMS 1100/. Minimalny zakres ochrony zaczyna się od wartości elementu struktury /ADABAS/, a kończy na pełnej strukturze danych /TOTAL, DMS 1100/. Podobnie jak z minimalnym kryterium dopuszczalności, którym jest dla większości SZBD typ operacji, a dla części /TOTAL, ROBOT, DMS 1100/ proces. Jednakże nawet te środki ochrony są zwłaszcza w warunkach systemów baz danych gospodarczych wykorzystywane w ograniczonym zakresie. Istnieje więc pilna konieczność prowadzenia działań mających na celu - zwłaszcza w warunkach krajowych - zmianę tego stanu rzeczy.

Literatura

/1/ J.P.Anderson: Information Security in a Multiuser Computer Environment. Advances in Computers, Academic Press, New York 1972 vol. 12.

/2/ A.Brandt: Funkcje administratora bazy danych i administratora zastosowań. CPiZI Warszawa 1981.

/3/ P.S.Brown: Data Privacy and Integrity: an Overview. Proceedings of 1971 ACM - SIGFIDET Workshop on Data Description, Access and Control, ACM San Diego 1971.

/4/ D.Clements: Fuzzy Models for Computer Security Metrics. University of California, Berkeley 1977 /praca doktorska/.

/5/ R.W.Conway, W.L.Maxwell, H.L.Morgan: On the Implementation of Security Measures in Information Systems. Communikations of the ACM 1972 no 4.

/6/ B.Davis: The Selection of Database Software. NCC Publications, Manchester 1977.

/7/ W.Diffie, M.E.Hellman: A Critique of the Proposed Data Encryption Standard. Communications of the ACM 1976 no 3.

/8/ M.Dyczkowski: Mechanizmy ochrony dostępu do danych w wybranych Systemach Zarządzania Bazą Danych. Prace Naukowe AE, Wrocław 1981 nr 183.

/9/ M.Dyczkowski: Metoda poziomów uprawnień jako środek selektywnej ochrony dostępu do danych w systemie informowania kierownictwa. Prace Naukowe AE,Wrocław 1982 nr 205.

/10/ A.C.Ellis: Analysis of Some Abstract Measures of Protection in Computer Systems. International Journal of Computer and Information Science 1978 no 3.

/11/ K.P.Eswaran, N.J.Gray, R.A.Lorie, I.L.Traiger: The Notions of Consistency and Predicate Locks in Data Base System. Communications of the ACM 1976 no 11.

/12/ G.C.Everest: Concurrent Update Control and Data Base Integrity. Proceedings of the IFIP TC-2 Working Conference on Data Base Management Systems, North Holland Amsterdam 1974.

/13/ H.M.Gladney, E.L.Worley, J.M.Myers:

Access Control Mechanism for Computing Resources. IBM Systems Journal 1975 no 3.

/14/ G.S.Graham, P.J.Denning: Protection - principles and practice. Spring Joint Computer Conference 1972.

/15/ P.Griffiths, B.Wade: An Authorization Mechanism for Relational Database Systems. ACM Transactions on Database Systems 1976 no 3.

/16/ M.A.Harrison, W.L.Ruzzo, J.D.Ullman: Protection in Operating Systems. Communications of the ACM 1976 no 8.

/17/ R.Hartson, D.K.Hsiao: A Semantic Model for Database Protection Languages. Proceedings of Second Very Large Data Base Conference, Brussels 1976.

/18/ G.E.Hemmings: Information Privacy in the Data Base Environment. Data Base Management. International Computer State of the Art. Report 15. INFOTECH Information 1973.

/19/ L.J.Hoffman: Poufność w systemach informatycznych. WNT Warszawa 1982.

/20/ D.K.Hsiao, D.S.Kerr, S.E.Madnick: Computer Security. Academic Press, New York 1979.

/21/ H.Jr.Katzan: Computer Data Security. Van Nostrand Reinhol Company, New York 1973.

/22/ J.T.Martin: Computer Data-Base Organization. Prentice-Hal Englewood Cliffs, New Jersey 1975.

/23/ J.T.Martin: Principles of Data-Base Management. Prentice-Hall, Englewood Cliffs, New Jersey 1976.

/24/ J.T.Martin: Security, Accuracy and Privacy in Computer Systems. Prentice-Hall, Englewood Cliffs, New Jersey 1973.

/25/ Z.Michalewicz: Data Base Security. IPI PAN Warszawa 1982.

/26/ R.Morris, K.Thompson: Password Security: A Case History. Communications of the ACM 1979 no 11.

/27/ NBS/ACM: Executive Guide to Computer Security. National Bureau of Standards and Association for Computing Machinery, Washington 1974.

/28/ Przegląd metod zarządzania zbiorami i terminalami dla potrzeb systemu działającego w czasie rzeczywistym na R-32. ZNB AE, Wrocław 1981.

/29/ J.H.Saltzer, M.D.Schroeder: The Protection of Information in Computer Systems. Proceedings of the IEEE 1975 no. 9.

/30/ E.H.Sibley: The Development of Data Base Technology. ACM Computing Surveys 1976 no 8.

/31/ W.Staniszkis: Mechanizmy ochrony danych w systemach zarządzania bazą danych. OBRI, Warszawa 1978.

/32/ Środki bezpieczeństwa i plany działań w sytuacjach awaryjnych. EPB Diebolda zeszyt 120. CPiZI Warszawa 1981.

/33/ D.C.Tsichritzis, F.M.Lochowsky: Data Base Management Systems, Academic Press, New York 1977.

/34/ B.J.Walker, I.F.Blake: Computer Security and Protection Structures. DHR Inc. 1977.

/35/ G.Wiederhold: Database Design. McGraw-Hill Book Company, New York 1977.

/36/ D.VanTassel: Computer Security Management. Prentice-Hall, Englewood Cliffs, New York 1972.

mgr JANUSZ GOCAŁEK
mgr inż. JACEK KLAUZIŃSKI
Instytut Technologii
i Konstrukcji Budowlanych
Politechnika Poznańska

KOMPILATOR JĘZYKA FORTRAN W SYSTEMIE OPERACYJNYM CROOK DLA MERY 400

W Instytucie Okrętowym Politechniki Gdańskiej opracowano system operacyjny CROOK dla Mery 400 [1], [2]. System ten zyskał uznanie użytkowników w wielu ośrodkach obliczeniowych na terenie całego kraju. Podstawowym mankamentem systemu był brak kompilatorów języków wyższego poziomu /z wyjątkiem języków BASIC i CEMMA/, w szczególności kompilatora języka FORTRAN. Kompilator języka FORTRAN [4] dla minikomputera Mera 400, pracującego w systemie operacyjnym CROOK, opracowano w Instytucie Technologii i Konstrukcji Budowlanych Politechniki Poznańskiej. Dla obsługi wejścia i wyjścia oraz biblioteki standardowej, za punkt wyjścia przyjęto język FORTRAN-1900 /3/.

XFOR /taką przyjęto nazwę/ jest kompilatorem jednoprzebiegowym. W wyniku kompilacji generowany jest kod w postaci rozkazów języka symbolicznego ASSM [5]. Assembler następnie dokonuje przekładu na postać binarną. Faza konsolidacji zawarta jest w tym samym przebiegu. Kompilator XFOR, w swej podstawowej wersji, zajmuje 16K słów PAO /pamięci operacyjnej/. Natomiast w wersji rozszerzonej o możliwość nakładkowania programów fortranowskich oraz korzystania ze zbiorów binarnych o dostępie bezpośrednim, zajmuje 20K słów PAO.

Podstawowe zalety kompilatora języka FORTRAN w systemie operacyjnym CROOK to:
- bardzo duża szybkość kompilacji,
- optymalny kod wynikowy,
- łatwość w obsłudze /kompilacja automatyczna/,
- możliwość dołączania wielu bibliotek,
- możliwość tłumaczenia programów fortranowskich na język Assemblera,
- możliwość pisania instrukcji w dowolnej kolumnie wiersza,
- programowa obsługa błędów przystawki zmiennoprzecinkowej /w przypadku wystąpienia podmiaru przyjmuje się wartość zerową/,
- wielodostępność /co wynika z właściwości systemu operacyjnego/.

Dyrektywy sterujące procesem kompilacji

Dyrektywy umożliwiają sterowanie przebiegiem kompilacji, zgodnie z ustalonym przez użytkownika tokiem obliczeń. Poszczególne zlecenia pozwalają na:
- deklarację strumieni wejściowego i listującego,
- deklarację nazwy zbioru binarnego,
- definiowanie opcji: listowania, śledzenia programu, badania przekroczenia zakresu tablic, numeracji wierszy podczas listowania,
- dołączanie dowolnych bibliotek użytkownika napisanych w języku symbolicznym ASSM,
- scalanie segmentów programu źródłowego /w języku FORTRAN/ zapisanych w różnych zbiorach,
- nakładkowanie określonych segmentów programu źródłowego,
- automatyczne wywołanie Assemblera.

Większość wyżej wymienionych dyrektyw ma charakter opcjonalny.

Informacja o języku

Język FORTRAN-CROOK oparty jest na implementacji języka FORTRAN-1900 /wersja podstawowa/ z nieznacznymi zmianami i odstępstwami od standardu ISO [6]:
- wszystkie instrukcje mogą występować w dowolnej kolumnie wiersza /od 1 do 72/, etykiety winny poprzedzać instrukcje i rozpoczynać się od dowolnej kolumny,
- dopuszcza się instrukcję "IF" logiczną typu arytmetycznego postaci:

IF (X) K1, K2

gdzie: alternatywa K1 jest wybierana, gdy X ma wartość "TRUE", a K2 gdy X ma wartość "FALSE",
- w instrukcji "DO" nie narzuca się żadnych ograniczeń na wartość parametrów: początkowego, końcowego i kroku, gdy parametry cyklu definiują cykl pusty, zostaje on pominięty,
- dopuszcza się w instrukcjach wejścia i wyjścia, by identyfikator specyfikacji wzorca był

etykietą lub pseudoetykietą "0" /zero/. Pseudoetykieta "0" nie jest związana z żadną instrukcją FORMAT i definiuje swobodną konwersję danych i wyników. Swobodna konwersja może być określona również przez definicję wzorca postaci IO, FO.O, EO.O itp.,
- wprowadzono instrukcję ASSEMBLER, która pozwala na umieszczenie w programie fortranowskim rozkazów w języku symbolicznym ASSM.

W zakresie składni pozostałych instrukcji, język ten nie różni się zasadniczo od języka FORTRAN-1900.

Biblioteki standardowe

Kompilator wyposażony jest w biblioteki standardowe, które przeszukiwane są na etapie kompilacji ASSEMBLEREM. Opracowano trzy typy bibliotek systemowych, niezbędnych dla uzyskania pełnego, gotowego modułu binarnego:
- bibliotekę zawierającą zestaw funkcji standardowych w zakresie liczb INTEGER i REAL, podprogramy obsługi wejścia i wyjścia, sygnalizacji błędów wykonania oraz śledzenia programu, jest ona przeszukiwana przy każdej kompilacji programu,
- bilbiotekę funkcji standardowych oraz działań na liczbach zespolonych /COMPLEX/.
- bibliotekę funkcji standardowych oraz działań na liczbach podwójnej precyzji /DOUBLE PRECISION/.

Dwie ostatnie biblioteki przeszukiwane są tylko wtedy, gdy w programie fortranowskim pojawi się odpowiednio deklaracja typu COMPLEX lub DOUBLE PRECISION. W trakcie przeszukiwania bibliotek wybierane są i dołączane do programu wynikowego tylko te funkcje, do których wystąpiło odwołanie w programie źródłowym. Przeszukiwanie to odbywa się w końcowej fazie kompilacji ASSEMBLEREM, co stwarza możliwość przekrywania funkcji standardowych przez funkcje deklarowane w programie użytkownika, tzn.: zdefiniowania funkcji o tej samej nazwie w ciele programu fortranowskiego. Zestaw funkcji standardowych /wewnętrznych oraz zewnętrznych /jest identyczny z zestawem funkcji standardowych języka FORTRAN-1900 /3/.

Kompilator języka FORTRAN-CROOK charakteryzuje się przede wszystkim bardzo dużą szybkością kompilacji. Czas kompilacji przykładowego programu zawierającego około 2000 rekordów fortranowskich /wraz z kompilacją Assemblerem i równoczesną konsolidacją/, przy automatycznej pracy, nie przekracza 4 minut. Daje to około 15-krotne skrócenie czasu kompilacji w stosunku do pracy w firmowym systemie operacyjnym SOM-3 /7/. Uzyskano również efekt średnio 2-krotnego skrócenia czasu wykonania programów binarnych w stosunku do tych samych programów wykonywanych w systemie SOM-3.

Kompilator XFOR eksploatowany jest od września 1982 r. w Instytutach Politechniki Poznańskiej oraz niektórych ośrodkach obliczeniowych w kraju.

Literatura

/1/ Z.Czerniak: System operacyjny CROOK-3 dla maszyny cyfrowej Mera 400 - opis użytkowania. Wydawnictwo Politechniki Gdańskiej, Gdańsk 1982.

/2/ Z.Czerniak, M.Nikodemski: System operacyjny CROOK dla Mery 400. Informatyka 1980, nr 6, s. 20-21.

/3/ FORTRAN - Opis języka, ELWRO, nr 13018, Wrocław, 1976.

/4/ J.Gocalek, J.Klauziński: Opis użytkowy kompilatora FORTRAN -CROOK. ITKB-PP, Poznań 1983.

/5/ W.J.Martin: Język symboliczny ASSM dla maszyny cyfrowej Mera 400 - podręcznik programowania. Wydawnictwo Politechniki Gdańskiej, Gdańsk 1982.

/6/ Programming language FORTRAN, ISO - recommendation, - 1539. ISO 1972.

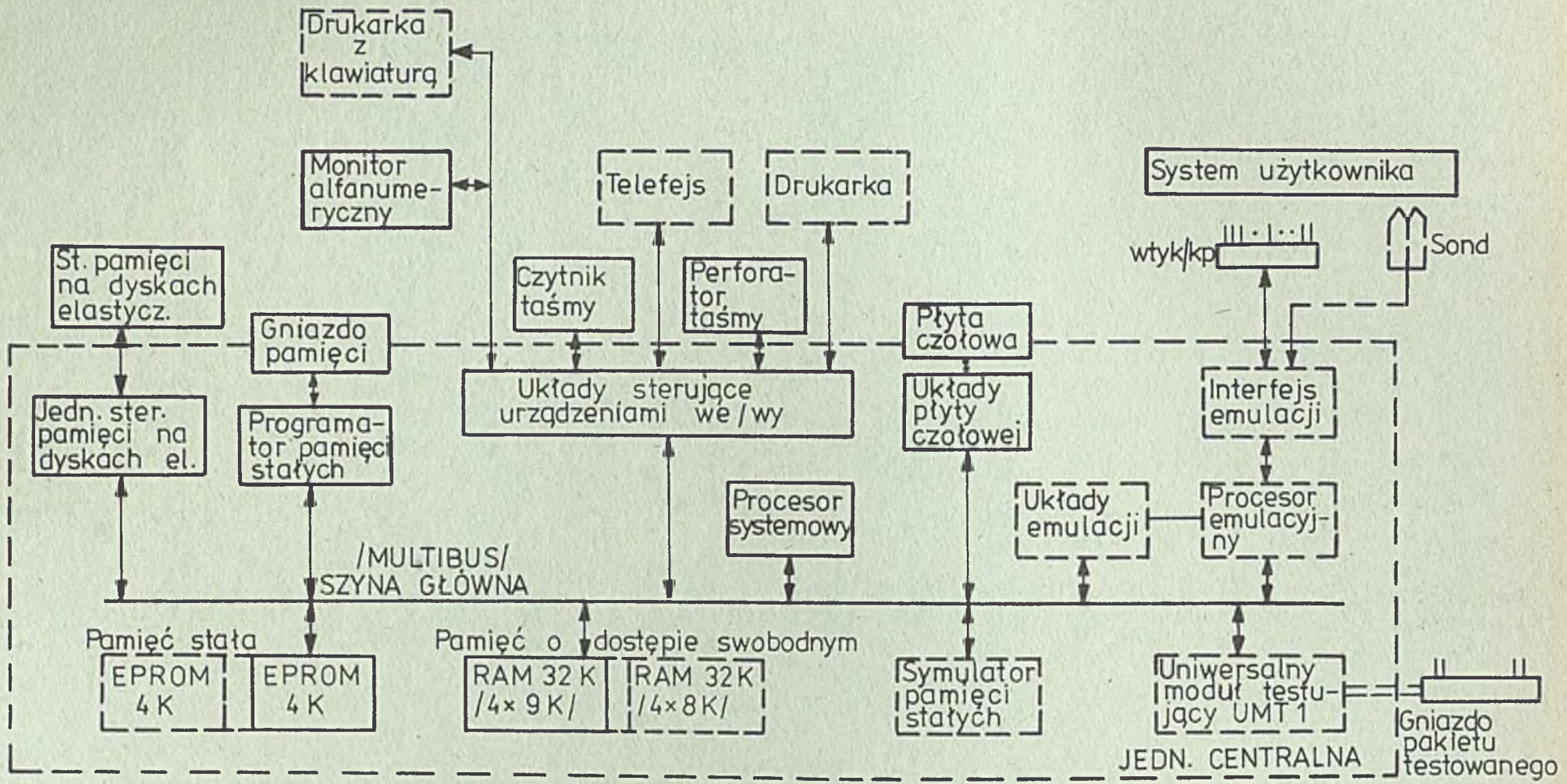
/7/ Opis eksploatacyjny systemu operacyjnego SOM-3. Dokumentacja techniczno-ruchowa, tom III, Warszawa 1979.

SPIS ARTYKUŁÓW

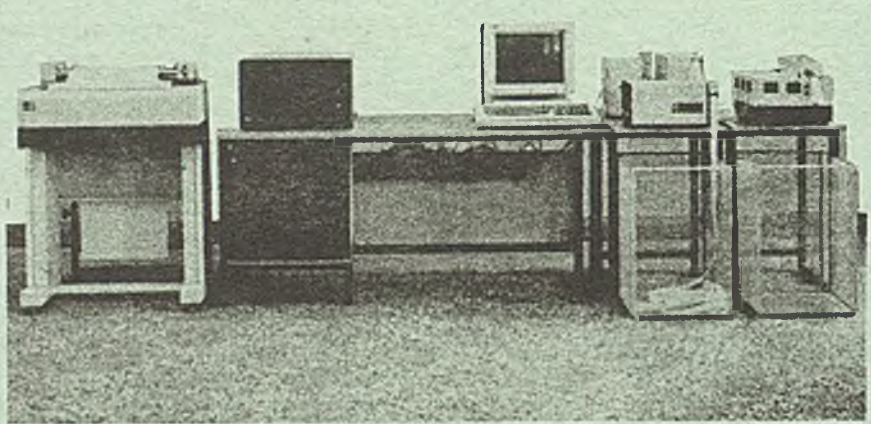
"POMIARY-AUTOMATYKA-KONTROLA"— nr 4-5 /1983

● ● ●

MIKROPROCESOROWY SYSTEM WSPOMAGANIA PROJEKTOWANIA



Opracowanie: Instytut Maszyn Matematycznych, ul. Krzywickiego 34, 02-078 Warszawa, tel. 21-84-41 w. 586, telex 813517



MERA