ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1980

Seria: INFORMATYKA z. 1 Nr kol. 653

Jan BRUSKI

Bogdan GRUDZIŃSKI

Ośrodek Elektronicznej Techniki Obliczeniowej
Politechnika Śląska

# O WYKORZYSTANIU STRUKTURALNEJ ORGANIZACJI JEDNORODNEJ PAMIĘCI MASZYNY CYFROWEJ I JEJ ZWIĄZKU Z SYSTEMEM STRUKTUR BINARNYCH

Streszczenie. W artykule przedstawiono potrzebę stosowania strukturalnej organizacji pamięci, a także istotę jednego ze sposobów tworzenia takiej organizacji. Pokazany został również jej związek z teorią struktur binarnych oraz teorią grafów.

Procesom przetwarzania informacji towarzyszy zawsze potrzeba przechowywania informacji. Podstawowym ośrodkiem przechowywania informacji w maszynie cyfrowej jest pamięć operacyjna. W ośrodku przechowywania informacji, a więc w pamięci maszyny cyfrowej, można tworzyć pewne obiekty zawierające określoną treść informacyjną, a także można te obiekty przekształcać, wykonując na nich pewne ściśle określone operacje. Rodzaj obiektów, jakie można budować w pamięci maszyny cyfrowej, zależy od tego, jak jest ta pamięć zorganizowana.

Pamięci operacyjne współcześnie produkowanych maszyn cyfrowych z reguły są jednorodne. Oznacza to, że istnieje bezpośredni dostęp do każdego, dowolnego miejsca pamięci. Kolejnym miejscom takiej pamięci są w jednolity sposób przyporządkowane tzw. adresy, interpretowane najczęściej jako nieujemne liczby całkowite. Zawartość każdego miejsca pamięci jest natychmiast dostępna przez podanie adresu (numeru) tego miejsca. Ze względu na liniowy porządek adresów poszczególnych miejsc pamięci organizację takiej pamięci określa się jako liniową.

Wydawałoby się, że tego rodzaju organizacja pamięci, połączona z bezpośrednim dostępem do każdego miejsca pamięci, jest najbardziej korzystna. W wielu przypadkach rzeczywiście tak jest.

Jeżeli tworzone w pamięci obiekty mają strukturę regularną, wówczas faktycznie najbardziej dla nich właściwym ośrodkiem przechowywania informacji jest pamięć liniowo uporządkowana o bezpośrednim dostępie. Także wtedy, gdy budowane obiekty są nieregularne, ale ich postać i wielkość są

z góry ściśle określone, można stosunkowo łatwo wykorzystywać liniową organizację pamięci [5].

Sytuacja zdecydowanie się zmienia wtedy, gdy tworzone i przetwarzane obiekty mają strukturę nieregularną i w dodatku ich konfiguracja i rozmiar nie są z góry znane, ale wielkości te ustala się dopiero w trakcie przetwarzania informacji. Może mieć to miejsce na przykład wtedy, gdy maszynę cyfrową stosuje się do tzw. przetwarzania danych lub gdy dokonuje się przetwarzania symboli (np. w procesach translacji) [6], [7]. Można oczywiście w takich procesach stosować wyłącznie obiekty regularne, lecz zachodzi wtedy potrzeba wykonywania pewnych nienaturalnych przekształceń rzeczywistych obiektów. Byłoby znacznie lepiej, gdyby można było stosować najbardziej ogólne obiekty, zwane strukturami danych i bez żadnych dodatkowych przekształceń, bezpośrednio lokalizować je w pamięci maszyny cyfrowej.

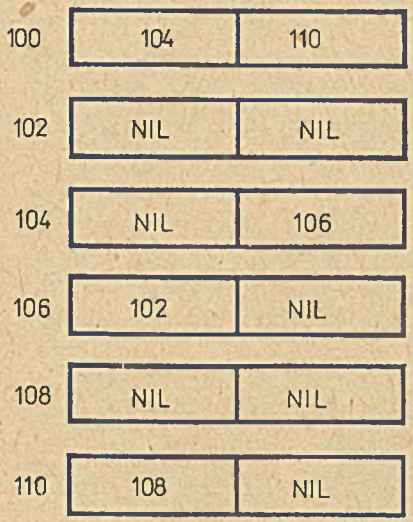
Istnieje jeszcze inny problem, którego rozwiązanie stwarza kłopoty w przypadku liniowej organizacji pamięci. Powstaje on wtedy, gdy wewnętrzna reprezentacja informacji w ośrodku przechowywania ma być niejednorodna. Jeżeli wszystkie używane pozycje informacyjne mają jednakowe i stałe długości, to wewnętrzne reprezentacje tych informacji będą miały również jednakowe długości i wówczas ich lokalizacja w pamięci o liniowej organizacji jest prosta i efektywna. Podobnie nie ma większego kłopotu, gdy istnieje kilka grup informacji, a w ramach każdej grupy długości pozycji informacyjnych są jednakowe. Często jednak trzeba przetwarzać takie informacje, których wewnętrzne reprezentacje wymagają różnej i czasem nieokreślonej liczby jednostek (bitów, znaków czy słów). W takich przypadkach znowu liniowa organizacja pamięci nie jest zbyt korzystna, natomiast doskonałe wyniki daje w tym względzie organizacja strukturalna [3].

Tworzenie strukturalnej organizacji pamięci jednorodnej może się odbywać w różny sposób [5]. Jeden z nich polega na tym, że całą pamięć dzieli się na pewną liczbę pól. Każde pole tworzy się ze ściśle określonej, jednakowej liczby miejsc pamięci (komórek). Najczęściej w skład poszczególnych pól wchodzą kolejne komórki pamięci (nie musi to jednak stanowić reguły). Dostęp do każdego pola uzyskuje się przez podanie adresu początkowej komórki pamięci zawartej w tym polu.

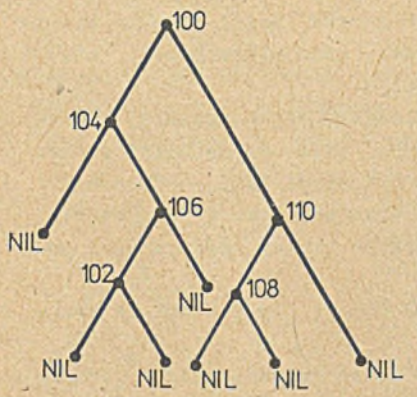
Wszystkie pola pamięci wiąże się z sobą tak, że każde z nich może być powiązane z dwoma innymi, dowolnymi polami. Wiązania pól tworzy się w taki sposób, że w każdym polu zapisuje się informacje określające, które dalsze pola są związane z polem rozpatrywanym. Praktycznie dokonuje się to przez umieszczenie w każdym polu dwóch adresów wskazujących, jakie pola są podporządkowane temu, które się rozpatruje.

Jeden z możliwych adresów używanych do tworzenia wiązań jest specjalnie wyróżniony - nie dotyczy on bowiem żadnego pola pamięci. Zwyczajowo oznacza się go symbolem NIL. Używa się go wówczas, gdy do jakiegoś pola nie dowiązuje się już następnego, podrzędnego pola. Symbol NIL umieszczo-

ny w dowolnym polu oznacza więc, że odpowiednie dowiązanie do tego pola nie istnieje.




Rys. 1. Przykład sposobu wiązania pól pamięci




Rys. 2. Przykład drzewa binarnego

Cechą charakterystyczną tworzonych wiązań jest to, że budując powiązania nie trzeba brać pod uwagę kolejnych pól (według porządku ich adresów), ale można wiązać z sobą dowolne pola, byle były one dotąd niewykorzystane. Na rys. 1 pokazano w symboliczny sposób przykład powiązania z sobą kilku pól pamięci.

Bardzo często wymaga się, aby sposób powiązań pól pamięci był hierarchiczny. Oznacza to, że polem podporządkowanym (dowiązanym) nie może być takie pole, które jest polem nadrzędnym (bezpośrednio lub pośrednio) w stosunku do rozpatrywanego pola. Tym bardziej żaden z adresów, które w rozpatrywanym polu określają dowiązania, nie może wtedy wskazywać na to właśnie pole.

Omówionemu sposobowi hierarchicznego wiązania pól pamięci odpowiada pewien zorientowany graf stanowiący tzw. drzewo binarne [3]. Drzewo binarne zawiera jeden wierzchołek zwany początkowym (lub korzeniem drzewa), do którego nie dochodzi żaden łuk, natomiast wychodzą z niego dwa łuki, pewną liczbę tzw. wierzchołków końcowych (zwanych czasem liśćmi drzewa), od których nie odchodzą żadne łuki oraz pewną liczbę wierzchołków pośrednich, które charakteryzują się tym, że do każdego z nich dochodzi jeden łuk, a odchodzą dwa łuki. Przykład drzewa binarnego został pokazany na rys. 2.

Zorientowany graf G można określić formalnie [4] jako system (N, E, g) taki, gdzie N jest zbiorem wierzchołków, E - zbiorem łuków, przy czym $N \cap E = \emptyset$, a g jest funkcją określoną następująco: $g : E \rightarrow N \times N$.

Jeżeli $e \in E$ i $g(e) = (n_1, n_2)$, to $n_1$ nazywa się wierzchołkiem początkowym łuku e, a $n_2$ jest wierzchołkiem końcowym łuku e. Mówi się też czasem, że wierzchołki $n_1$ i $n_2$ są związane łukiem e.

Jeżeli zbiór wierzchołków N jest pusty, to otrzymuje się pusty graf.

Drzewem zorientowanym lub, krótko, drzewem T nazywa się albo pusty graf (jest to wtedy drzewo puste), albo dwójkę $(G, n_0)$, gdzie G jest niepustym grafem, a $n_0 \in N$ jest wierzchołkiem początkowym drzewa T.

Drzewo T posiada następujące własności:

- liczba wierzchołków jest skończona,
- istnieje jeden i tylko jeden taki wierzchołek (oznaczony $n_0$), który nie jest wierzchołkiem końcowym dla żadnego łuku należącego do E. Wierzchołek $n_0$ nazywa się wierzchołkiem początkowym drzewa T lub korzeniem drzewa,
- dla dowolnego wierzchołka $n \neq n_0 \wedge n \in N$ istnieje dokładnie jeden łuk $e \in E$ taki, że $g(e) = (n', n)$, gdzie $n' \in N$. Mówimy, że wierzchołek $n'$ bezpośrednio poprzedza wierzchołek n (jest nadrzędny w stosunku do n).
- dla dowolnego wierzchołka $n = n_k \neq n_0$ istnieje droga: $e_1, e_2, \ldots, e_k$ taka, że: $f(e_i) = (n_{i-1}, n_i)$, $i = 1,2,\ldots,k$. Oznacza to, że drzewo stanowi graf spójny.

Każde niepuste drzewo zawiera co najmniej jeden wierzchołek n taki, że nie istnieje łuk e i wierzchołek $n''$, dla których $g(e) = (n, n'')$. Wierzchołki o tej własności nazywa się wierzchołkami końcowymi drzewa lub liśćmi drzewa. Zbiór $N_1$ wszystkich wierzchołków końcowych drzewa jest podzbiorem zbioru N. Jeżeli $N = \{n_0\}$, to $N_1 = N = \{n_0\}$.

Żadne dowolne drzewo nie zawiera takiego wierzchołka n, dla którego istniałaby droga $e_1, e_2, \ldots, e_m$ taka, że: $f(e_1) = (n, n_1)$, $f(e_2) = (n_1, n_2)$, $\ldots$, $f(e_m) = (n_{m-1}, n)$. Oznacza to, że drzewo nie zawiera zamkniętych dróg (obwodów).

Jeżeli, z wyjątkiem wierzchołków końcowych drzewa, każdy dowolny wierzchołek $n \in N$ zawsze bezpośrednio poprzedza dwa inne wierzchołki, wtedy tak zbudowane drzewo nazywa się drzewem binarnym.

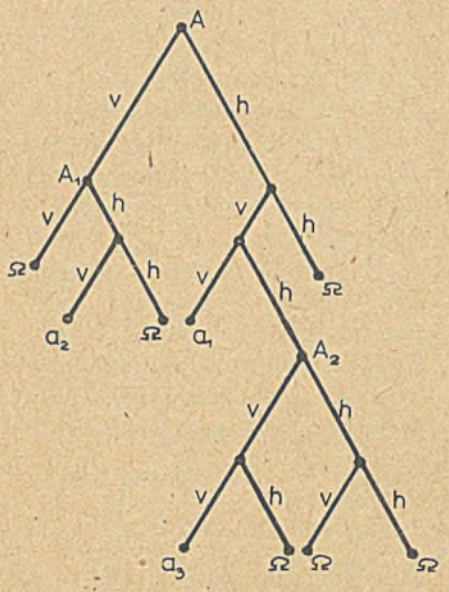
Drzewo binarne może posłużyć do przedstawienia opisanej wcześniej strukturalnej organizacji pamięci. W tym celu trzeba tylko wierzchołki drzewa przyporządkować poszczególnym polom pamięci, natomiast łuki - adresom dowiązań zawartym w rozpatrywanych polach. Przykładowo, rys. 2 przedstawiający drzewo binarne może stanowić graficzny obraz wiązań pól pamięci pokazanych na rys. 1.

Drzewo binarne będzie zawierało więcej informacji, gdy zostanie oznakowane.

Oznakowanym drzewem binarnym $T_L$ jest albo puste drzewo, albo trójka $(T, L, f)$, gdzie $T = (N, E, g, n_0)$ jest drzewem binarnym, L - niepustym zbiorem oznaczeń nie przecinającym się ze zbiorami N i E, a $f: E \cup N \to L$ jest funkcją, która wierzchołkom i łukom drzewa binarnego przyporządkowuje oznaczenia ze zbioru L.

Funkcja f posiada taką właściwość, że gdy dowolne wierzchołki drzewa $n_1$ i $n_2$ związane są z pewnym innym wierzchołkiem n za pośrednictwem łuków $e_1$ i $e_2$, czyli, gdy: $g(e_1) = (n, n_1)$ i $g(e_2) = (n, n_2)$, to $f(e_1) \neq f(e_2)$.

Oznakowane drzewa binarne mogą być użyte do ilustracji obiektów stanowiących struktury binarne [1].

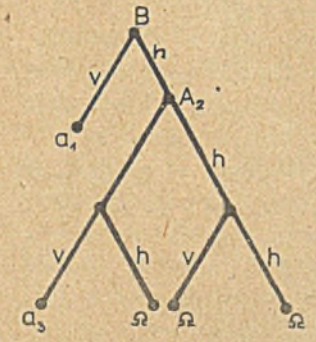



Rys. 3. Drzewo binarne stanowiące graficzny obraz struktury binarnej A

Jeżeli trójka $(\mathcal{K}, S, o)$ jest systemem struktur binarnych, gdzie $\mathcal{K}$ jest zbiorem obiektów tego systemu, S zbiorem selektorów binarnych, natomiast o operacją określoną nad tym systemem, wówczas każdy obiekt należący do zbioru $\mathcal{K}$ może być przedstawiony za pomocą klasy równoważności oznakowanych drzew binarnych. Wystarczy przyjąć następujące przyporządkowania:

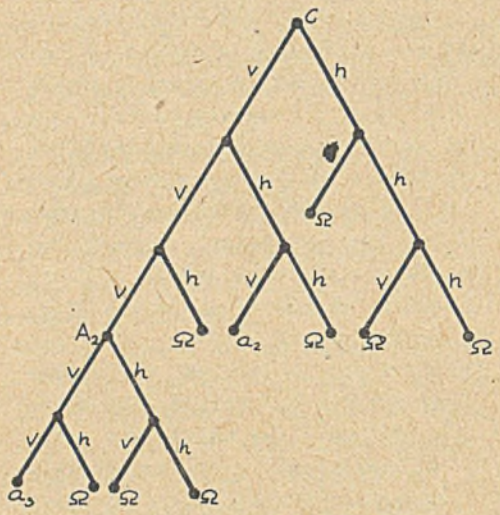
- Obiektowi pustemu $\Omega \in \mathcal{K}$ odpowiada drzewo puste.
- Obiektom należącym do zbioru atomów $\mathcal{A} \subset \mathcal{K}$ odpowiada klasa drzew z których każde składa się tylko z wierzchołka początkowego (korzenia). Wierzchołek ten jest oznakowany za pomocą symbolu a $(a \in \mathcal{A})$.
- Każdemu obiektowi $A = \{(v, A_v), (h, A_h)\}$ ze zbioru struktur binarnych $\mathcal{B} \subset \mathcal{K}$ odpowiada klasa równoważności drzew binarnych, które posiadają następujące własności:

Korzeń drzewa jest oznaczony symbolem A $(A \in \mathcal{B})$. Łuki wychodzące z wierzchołka początkowego drzewa binarnego są oznaczone kolejno v i h $(v, h \in S)$. Jeżeli dowolny z obiektów $A_v$ i $A_h$ jest atomem $a \in \mathcal{A}$, wówczas wierzchołek drzewa binarnego, związany z wierzchołkiem początkowym za pomocą łuku oznaczonego odpowiednio v lub h, oznacza się symbolem tego atomu. Gdy natomiast tak nie jest i jeden z tych obiektów (lub obydwa) jest strukturą binarną, to odpowiedni wierzchołek oznacza się przyjętą nazwą tej struktury (lub nie oznacza się go wcale). Znając postać tej struktury, można potraktować odpowiadający jej wierzchołek drzewa binarnego jako korzeń poddrzewa i stosować wyżej podane zasady dla oznakowania następnego fragmentu drzewa. Stosując tę procedurę wielokrotnie, można zaopatrzyć w oznaczenia wszystkie łuki i wierzchołki drzewa binarnego.




Rys. 4. Drzewo binarne odpowiadające strukturze binarnej B

Zgodnie z teorią struktur binarnych obiekt pusty Ω posiada dwoistą naturę: może być traktowany jako struktura binarna lub jako atom. Na gruncie drzew binarnych ta dwoistość wyraża się tym, że obiektowi Ω może odpowiadać albo drzewo puste albo drzewo złożone tylko z wierzchołka początkowego. Jeżeli natomiast obiekt Ω stanowi składową pewnej struktury binarnej, wówczas odpowiada mu wierzchołek końcowy (liść) drzewa binarnego.




Rys. 5. Drzewo binarne stanowiące obraz struktury binarnej C

Na rys. 3 zostało pokazane drzewo binarne oznakowane zgodnie z podanymi zasadami, a odpowiadające przykładowej strukturze binarnej określonej analitycznie za pomocą zapisów:

$$A = \{(v, A_1), (h, \{(v, \{(v, a_1), (h, A_2)\}), (h, \Omega)\})\}$$

$$A_1 = \{(v, \Omega), (h, \{(v, a_2), (h, \Omega)\})\}$$

$$A_2 = \{(v, \{(v, a_3), (h, \Omega)\}), (h, \{(v, \Omega), (h, \Omega)\})\}$$

Za pomocą drzew binarnych można ilustrować podstawowe operacje, jakie wykonuje się na strukturach binarnych, a mianowicie operację wyboru i operację konstrukcji. Na rys. 4 zostało pokazane drzewo binarne odpowiadające obiektowi powstałemu z obiektu użytego w poprzednim przykładzie po zastosowaniu funkcji wyboru: $B = v \circ h \circ A = \{(v, a_1), (h, A_2)\}$.

Na rys. 5 pokazano natomiast drzewo binarne odpowiadające obiektowi, jaki powstanie po trzykrotnym użyciu funkcji konstrukcji [1]:

$$C_1 = \mathcal{K}(A; <(v \circ v \circ v) : A_2>) =$$

$$= \{(v, \{(v, \{(v, A_2), (h, \Omega)\}), (h, \{(v, a_2), (h, \Omega)\})\}), (h, \{(v, \{(v, a_1), (h, A_2)\}), (h, \Omega)\})\}$$

$$C_2 = \mathcal{K}(C_1; <h : A_2>) =$$

$$= \{(v, \{(v, \{(v, A_2), (h, \Omega)\}), (h, \{(v, a_2), (h, \Omega)\})\}), (h, A_2)\}$$

$$C = \mathfrak{K}(C_2;\ <(v \circ h) : \Omega>) =$$

$$= \Big\{(v,\ \Big\{(v,\ \Big\{(v,\ A_2),\ (h, \Omega)\Big\}),\ (h,\ \Big\{(v,\ a_2),\ (h, \Omega)\Big\})\Big\}),$$

$$(h,\ \Big\{(v, \Omega),\ (h,\ \Big\{(v, \Omega),\ (h, \Omega)\Big\})\Big\})\Big\}$$

Ogólne struktury danych można ilustrować za pomocą oznakowanych drzew [8]. Zasada oznaczania drzew jest podobna do stosowanej w przypadku drzew binarnych. Łuki oznacza się symbolami selektorów, natomiast wierzchołki nazwami obiektów.

Wykorzystując przyporządkowanie oznakowanych drzew ogólnym strukturom danych (tutaj sposób tego przyporządkowania jest intuicyjny), można za pomocą drzewa przedstawić każdą strukturę.


Rys. 6. Drzewo odpowiadające strukturze danych $\psi(C)$

Na rys. 6 pokazano przykładowo drzewo odpowiadające obiektowi $\psi(C)$ otrzymanemu w wyniku transformacji struktury binarnej C do systemu ogólnych struktur danych.

Struktury binarne i określone nad nimi operacje mogą być użyte do zapisu i modelowania wiązań pól strukturalnej organizacji pamięci maszyny cyfrowej. Odwrotnie, pola utworzone w jednorodnej pamięci i wiązania tych pól za pomocą adresów mogą stanowić realizację struktur binarnych w maszynie.

Adresy określające wiązania pól pamięci odpowiadają selektorom binarnym, natomiast hierarchiczna struktura pól odpowiada obiektom systemu struktur binarnych.

Atomy w takiej realizacji mogą stanowić informacje zawarte w pojedynczych polach, bądź też w wielu polach z sobą związanych i tworzących listy pól.

Obiektowi pustemu $\Omega$ odpowiada pole nie istniejące w pamięci. Jeżeli obiekt ten będzie interpretowany jako struktura binarna, wtedy w pamięci maszyny cyfrowej nie będzie pola, które byłoby odpowiednikiem tego obiektu. Podobna sytuacja występuje wtedy, gdy obiekt pusty interpretuje się jako atom. Wówczas jednak będą istniały dowiązania do pola, które odpowiada obiektowi $\Omega$ (będą to wyróżnione adresy NIL). Stwarza to możliwość budowy w pamięci maszyny cyfrowej "szkieletu" struktury binarnej, który stanowi niejako przygotowanie struktury binarnej do przechowywania informacji użytkowych. Określa się wtedy bowiem tylko sposób powiązania poszczególnych pól, natomiast nie ma potrzeby równoczesnego dowiązania tych pól, które będą zawierały informacje odpowiadające obiektom elementarnym.

Strukturalna organizacja pamięci ma sens jedynie wówczas, gdy zapewni się jej odpowiednią elastyczność. Przede wszystkim musi istnieć możliwość tworzenia w pamięci dowolnej liczby struktur danych o dowolnej konfiguracji (oczywiście w ramach dostępnej pojemności pamięci).

Z drugiej strony organizacja pamięci winna zapewniać w miarę optymalne, "oszczędne" wykorzystanie wszystkich dostępnych miejsc pamięci. Niebagatelną również sprawą jest stworzenie warunków na to, by szybkość przetwarzania struktur danych w maszynie cyfrowej nie była zbyt mała. Dwa ostatnie wymagania stoją zazwyczaj w sprzeczności do siebie. "Oszczędna" gospodarka pamięcią maszyny cyfrowej wymaga zwykle wykonania większej liczby operacji pomocniczych, a to zmniejsza efektywną szybkość przetwarzania.

Najbardziej celowy wydaje się więc kompromis polegający na tym, że w normalnych warunkach, gdy kwestia gospodarki pamięcią nie jest krytyczna, dąży się do uzyskania możliwie największej szybkości przetwarzania.

Natomiast w sytuacji, gdy przetwarza się duże zespoły danych lub gdy program przetwarzający ma dużą objętość, włącza się dodatkowo mechanizmy pozwalające prowadzić oszczędną gospodarkę pamięcią. Wtedy bowiem optymalne wykorzystanie pamięci jest zadaniem pierwszoplanowym i musi być zrealizowane, by w ogóle osiągnąć cel, jaki został postawiony przed programem przetwarzającym. Mniejsza szybkość przetwarzania jest wtedy po prostu złem koniecznym, z którym trzeba się pogodzić. (Nie oznacza to oczywiście, że rezygnuje się zupełnie z optymalizacji działania ze względu na szybkość przetwarzania).

Opisany wcześniej sposób tworzenia strukturalnej organizacji pamięci w znacznym stopniu pozwala na spełnienie określonych wyżej wymagań.

Wszystkie pola pamięci zawierają jednakową liczbę miejsc. Dostęp do poszczególnych pól odbywa się przez podanie ich adresów początkowych. Budowa wszystkich pól może być jednakowa - każda pozycja informacyjna może mieć stałe położenie w polu. Cechy te zapewniają możliwość uzyskania znacznej szybkości operacyjnej. Można bowiem wykorzystywać właściwość bezpośredniego dostępu do każdego pola pamięci, a oprócz tego eliminuje się konieczność czasochłonnego badania zawartości pól po to, by otrzymać wiadomości o strukturze i charakterze zawartych w każdym polu pozycji informacyjnych.

Wielkość pól może być odpowiednio dobrana do wymaganej treści informacyjnej. Uzyskuje się wtedy maksymalne nasycenie pól zawartością informacyjną. Każde pole może być wówczas optymalnie wykorzystane.

Pewne pozycje informacyjne pól mogą być w razie potrzeby zróżnicowane ze względu na interpretację ich zawartości. Przykładowo, w pewnych wypadkach można je wykorzystywać do zapisu adresów dowiązań, a w innych wypadkach do zapisu wartości atomów. Rozróżnienie charakteru informacji następuje wtedy przez proste badanie wartości pewnych wskaźników zawartych w polach.

Przyjęcie omówionego sposobu budowy pól i organizacji ich wiązań nie stwarza również problemów, gdy istnieje potrzeba różnicowania atomów. Jeżeli wewnętrzna reprezentacja atomu mieści się w przeznaczonej dla niej części pola pamięci, wówczas oczywiście wystarczy dowiązać to pole do pozostałych pól tworzących strukturę danych.

Jeżeli reprezentacja atomu posiada większą długość, wówczas trzeba ją podzielić na pewną ilość części, każdą przechowując w jednym polu, wszystkie zaś pola trzeba związać, tworząc listę pól, którą dołącza się do odpowiedniej struktury. Uzyskuje się wtedy dostęp nie tylko do całego atomu, ale także do dowolnych jego fragmentów. W polu bezpośrednio poprzedzającym pola z przechowaną wartością atomu można dodatkowo umieścić informację stanowiącą specyfikację atomu.

Oczywiście można sobie wyobrazić inny sposób tworzenia strukturalnej organizacji pamięci i to w sposób bardziej optymalny ze względu na wykorzystanie miejsc pamięci [5]. Korzyści uzyskane przez optymalizację wykorzystania pamięci mogą jednak nie zrekompensować strat, jakie wynikną skutkiem zmniejszenia efektywności przetwarzania. Regularny sposób budowy pól pamięci powoduje wprawdzie, że stopień wykorzystania pamięci może być nieco mniejszy niż w niektórych innych systemach, ale za to otrzymuje się cały szereg innych, wyraźnych korzyści. Przede wszystkim istnieje bezpośredni związek omawianej organizacji z systemem struktur binarnych, a za pomocą struktur binarnych można przedstawiać dowolne obiekty systemu ogólnych struktur danych. Oprócz tego otrzymuje się bardzo przejrzysty obraz pamięci. Sposób korzystania z tworzonych struktur jest prosty i odznacza się dużą systematyką. Dzięki temu można uzyskiwać znaczne szybkości w trakcie przetwarzania.

Omawiany sposób budowy i wiązania pól sprzyja również stosowaniu mieszanej organizacji pamięci: liniowo-strukturalnej. Polega ona na tym, że w pamięci maszyny cyfrowej tworzy się dwa rozłączne obszary: jeden o organizacji liniowej, drugi o organizacji strukturalnej. Wszystkie obiekty regularne lokalizuje się w obszarze zorganizowanym liniowo, natomiast struktury nieregularne w pozostałej części pamięci. Daje to możliwość pełnego wykorzystania szybkości operacyjnej maszyny cyfrowej. Szybkość przetwarzania w warunkach organizacji strukturalnej jest zawsze mniejsza niż w przypadku liniowej organizacji pamięci. W tym jednak przypadku z pamięci zorganizowanej strukturalnie korzysta się tylko wtedy, gdy jest to konieczne. Aby możliwie najlepiej wykorzystać pamięć, jej podział na dwa obszary może być elastyczny. W zależności od potrzeb, jakie wynikną w trakcie przetwarzania informacji, wielkość jednego obszaru może się zwiększać kosztem drugiego.

Taki sposób organizacji pamięci zastosowano w systemie przetwarzania struktur danych w języku ALGOL 1900, którego opracowanie podjęto w Politechnice Śląskiej. Niniejszy artykuł stanowi wprowadzenie do opisu tego systemu.

LITERATURA

[1] Bruski J.: Struktury binarne i ich własności. Zeszyty Naukowe Pol.Śl., seria "Informatyka" z. 1.

[2] Bruski J.: Zbiory charakterystyczne struktur binarnych. Zeszyty Naukowe Pol.Śl., seria "Informatyka" z. 1.

[3] Berztiss A.T.: Data Structures. Theory and Practice. Academic Press, New York 1975.

[4] Korzan B.: Elementy teorii grafów i sieci. Metody i zastosowania. WNT, Warszawa 1978.

[5] Turski W.M.: Struktury danych. WNT, Warszawa 1976.

[6] Dahl O.J., Dijkstra E.W., Hoare C.A.R.: Structured Programming. Academic Press, London 1972.

[7] Ed. Genuys F.: Programming Languages. Academic Press, London 1976.

[8] Ollogren A.: Definition of Programming Languages by Interpreting Automata. Academic Press, London 1974.

О ИСПОЛЬЗОВАНИИ СТРУКТУРНОЙ ОРГАНИЗАЦИИ ОДНОРОДНОЙ ПАМЯТИ
ВЫЧИСЛИТЕЛЬНОЙ МАШИНЫ И ЕЁ СВЯЗИ
С СИСТЕМОЙ БИНАРНЫХ СТРУКТУР

Р е з ю м е

В статье рассматривается необходимость применения структурной организации памяти, а также суть одного из способов образования такой организации. Приводится тоже её связь с теорией бинарных структур и теорией графов.

ON EXPLOITATION OF STRUTURAL ORGANIZATION
OF HOMOGENEOUS COMPUTER'S MEMORY AND ITS RELATIONSHIP
WITH THE SYSTEM OF BINARY STRUCTURES

S u m m a r y

Necessity of usage of structural organization of memory and essence of a certain method of creating this organization were presented in the paper. Relation between this organization and theory of binary structures and theory of graphs were also shown.