Instytut Cybernetyki Technicznej
Politechniki Wrocławskiej

Ćwiczenia laboratoryjne
z "Teorii automatów"

Temat ćwiczenia :
Komputerowa realizacja automatów skończonych

Opracowali:
prof.dr hab.inż. Jan Kazimierczak
mgr inż. Dariusz Banasiak

Wrocław 1992

Ćwiczenie laboratoryjne z teorii automatów

TEMAT ĆWICZENIA: KOMPUTEROWA REALIZACJA AUTOMATÓW SKOŃCZONYCH.

1. CEL ĆWICZENIA

Celem ćwiczenia jest praktyczne zapoznanie się ze sposobem programowej realizacji na komputerze lub mikrokomputerze automatów skończonych (na przykładzie automatów typu Moore'a).

2. PROGRAM ĆWICZENIA

1. Zapoznanie się z metodą transformacji formalnego modelu automatu skończonego na model dogodny do zapamiętania w pamięci komputera.
2. Wprowadzenie z pulpitu komputera do pamięci komputera danych, które będą reprezentować w pamięci komputera model formalny automatu.
3. Zapoznanie się z programem działającym na elemantach wprowadzanego do pamięci modelu formalnego automatu.
4. Testowanie wprowadzonego do pamięci programowego wariantu automatu o małej liczbie stanów wewnętrznych.
5. Opracowanie wyników testowania programowego wariantu automatu skończonego o małej liczbie stanów wewnętrznych.
6. Sprawdzenie działania programowego wariantu automatu Moore'a o dużej liczbie stanów wewnętrznych.

3. WIADOMOŚCI PODSTAWOWE

3.1. Wstęp do programowej implementacji automatów skończonych

Jednym z głównych działów wchodzących w zakres nauki o komputerach (z jęz. angielskiego computer science) jest teoria automatów. Przedmiotem rozważań w teorii automatów są automaty skończone. Automat skończony definiuje się jako model formalny

(matematyczny) dyskretnego systemu lub procesu przebiegającego w dyskretnych chwilach czasu [2]. Zgodnie z tą definicją automat skończony jest pewną abstrakcją. Abstrakcję tą jednak można urealnić budując odpowiedni układ funkcjonalny, którego zachowanie się będzie zgodne z reprezentującym go modelem matematycznym. Na przykład, automat skończony można wykonać jako układ elektroniczny (hardware) lub jako program na komputer (software). Każdy układ sekwencyjny w strukturze fizycznej komputera można rozpatrywać jako pewien automat skończony, podobnie każdy program składowy systemu operacyjnego komputera (software) można rozpatrywać - przy wykonywaniu tego programu - jako pewien automat skończony.
W nauce o komputerach dobrze znany jest w odniesieniu do automatów skończonych fakt istnienia dualizmu między strukturą fizyczną komputera (hardware) i jego oprogramowaniem (software) [6]. Wybór hardware'owej lub software'owej realizacji automatu skończonego zależy od funkcji jaką ten automat będzie spełniał w strukturze funkcjonalnej komputera (jak również mikrokomputera).

Ponieważ przedmiotem prezentowanego ćwiczenia laboratoryjnego jest programowa realizacja automatu skończonego niżej przedstawimy ogólne zasady tej realizacji.

Jak wiadomo działanie automatu skończonego jest jednoznacznie określone grafem automatu nazywanego często grafem przejść (z jęz. ang. transition diagram). Przy programowej realizacji automatu skończonego jego graf powinien być zapamiętany w pamięci komputera. Aby można było tego dokonać graf automatu powinien być przekształcony na odpowiednią postać symboliczną dogodną do zapamiętania w pamięci komputera. Ponieważ problematyka ćwiczenia dotyczy programowej realizacji automatu Moore'a w pierwszej kolejności rozpatrzona zostanie transformacja grafu tego automatu na postać symboliczną.

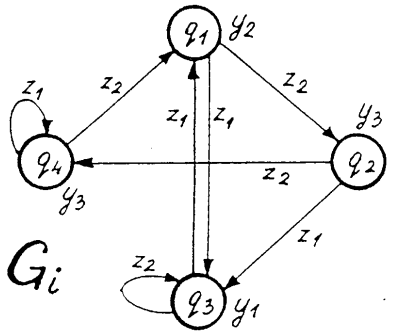
3.2. Postać symboliczna grafu automatu i jej reprezentacja w pamięci komputera

Punktem wyjścia do programowej realizacji automatu Moore'a jest transformacja grafu automatu na wyrażenie symboliczne.

Załóżmy, że zadany jest automat Moore'a reprezentowany "szóstką" $<Z,Q,Y,\phi,\psi,q_2>$:

gdzie $Z = \{z_1,z_2\}$ - alfabet wejściowy,
$Y = \{y_1,y_2,y_3\}$ - alfabet wyjściowy,
$Q = \{q_1,q_2,q_3,q_4\}$ - zbiór stanów wewnętrznych,
ϕ - funkcja przejść $q(t+1) = \phi\,[q(t),z(t)]$,
ψ - funkcja wyjść $y(t) = \psi\,[q(t)]$,
q_2 - stan początkowy automatu.

Przyjmujemy, że działanie tego automatu określone jest przez graf przedstawiony na rys.1. Graf ten, oznaczony symbolem G_i jest transformowany na wyrażenie symboliczne G_i^+ (zgodnie z algorytmem podanym w pracach [3],[4]).



Rys.1. Przykładowy graf automatu Moore'a.

Wyrażenie G_i^+ opisujące graf G_i z rysunku 1 ma postać:

$$G_i^+ = {}^0(q_2{}^1(z_1q_3{}^2(z_1q_1{}^3(z_2q_2, z_1q_3)^3, z_2q_3)^2, z_2q_4{}^2(z_2q_1, z_1q_4)^2)^1)^0 \quad (1)$$

W przedstawionym wyrażeniu symbol q_r $(r=1,2,..)$ stojący przed nawiasem otwierającym ${}^k($ - gdzie $k=1,2,..$ jest indeksem nawiasu - reprezentuje wierzchołek grafu, natomiast term typu $..q_r{}^k(z_jq_s..$ reprezentuje krawędź z_j wychodzącą z wierzchołka q_r i prowadzącą do wierzchołka q_s.

Wyrażenie G_i^+ (1) nie reprezentuje w pełni grafu G_i, ponieważ w wyrażeniu G_i^+ nie uwidocznione są przyporządkowania wierzchołkom q_r sygnałów wyjściowych y_i. Stąd też, aby uzyskać kompletną reprezentację grafu G_i, wyrażeniu G_i^+ należy przyporządkować tablicę $\overline{Q}_i(q_r \longleftrightarrow y_j)$.

Tablica ta ma postać:

q_r	y_j
q_1	y_2
q_2	y_3
q_3	y_1
q_4	y_3

W celu zapamiętania symbolicznej reprezentacji grafu G_i w pamięci komputera, wyraźnie G_i^+ jest przekształcane tak, aby za dowolnym termem typu $q_r{}^k($ w wyrażeniu G_i^+ znalazły się termy typu $z_j q_s$ reprezentujące wszystkie krawędzie wychodzące z danego wierzchołka q_r. Produktem tego przekształcenia jest wyrażenie symboliczne G_i^{++} o postaci:

$$G_i^{++}=0(q_2{}^1(z_2q_4,z_1q_3{}^2(z_2q_3,z_1q_1{}^3(z_2q_2,z_1q_3)^3)^2),q_4{}^1(z_2q_1,z_1q_4)^1)0 \qquad (2)$$

Wyrażenie G_i^{++} może być wprowadzone z pulpitu komputera (lub mikrokomputera) do jego pamięci w takiej postaci indeksowanej jaka pokazana została w wyrażeniu (2) lub w postaci indeksowanej stosowanej w językach programowania. W pierwszym przypadku wymagany jest odpowiedni program wprowadzający wyrażenie G_i^{++}, przy czym w tym przypadku wprowadzenie tego wyrażenia z pulpitu jest uciążliwe dla użytkownika. W drugim przypadku wyrażenie G_i^{++} wprowadzane jest do pamięci w następującej postaci:

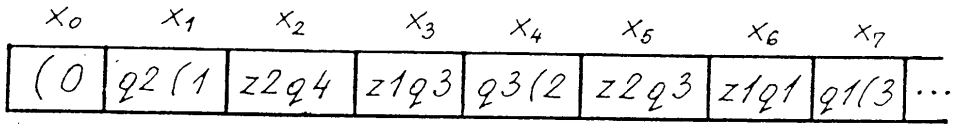
$$\tilde{G}_i^{++}=(0q2(1z2q4,z1q3(2z2q3,z1q1(3z2q2,z1q3)3)2)1,q4(1z2q1,z1q4)1)0 \qquad (3)$$

Do zapamiętania wyrażenia $\tilde{G}_i^{++}$ wydzielona jest liniowa sekwencja słów pamięci komputera. W słowach tych zapamiętywane są termy wyrażenia $\tilde{G}_i^{++}$ w takiej kolejności w jakiej występują one w wyrażeniu $\tilde{G}_i^{++}$. Termami wyrażenia G_i^{++} i $\tilde{G}_i^{++}$ są:

- nawias otwierający początkowy $0($ = (0 ;
- symbol q_r przed nawiasem otwierającym $q_r{}^k($ = qr(k ;
- para symboli $z_j q_s$, która w notacji wyrażania $\tilde{G}_i^{++}$ ma postać zjqs, na przykład z2q4 ;

- nawias zamykający $)^k =)k$.

Lokalizacja wyrażenia $\tilde{G}_i^{++}$ w pamięci komputera (mikrokomputera) przedstawia się tak, jak to pokazano na rys.2, gdzie dowolny symbol x_j oznacza adres komórki, w której przechowywany jest dany term. Analizując rysunek 2 łatwo zauważyć, że nie występuje tam przecinek, ponieważ położenie przecinka w $\tilde{G}_i^{++}$ rozpoznawane jest przez odpowiadni program, który w trakcie programowej realizacji rozpatrywanego automatu Moore'a działa na wyrażeniu $\tilde{G}_i^{++}$.



Rys.2. Rozmieszczenie elementów wyrażenia $\tilde{G}_i^{++}$ w pamięci i ich adresacja.

Jak już nadmieniono wyżej, działanie automatu skończonego jest jednoznacznie określone przez jego graf. Symboliczną reprezentacją grafu automatu, abstrahując od programowej realizacji automatu, jest wyrażenie G_1^+ i tablica $\bar{Q}_i(q_r \longleftrightarrow y_j)$. Przy programowej implementacji automatu reprezentacją grafu automatu w pamięci komputera jest wyrażenie $\tilde{G}_i^{++}$ (3) oraz tablica $\tilde{Q}_i$ będąca modyfikacją tablicy $\bar{Q}_i$ przez przyporządkowanie termom $q_r^k($ występującym w wyrażeniu G_i^{++} (2) adresów tych komórek pamięci, w których te termy będą pamiętane.
Tablica $\bar{Q}_i$ odpowiadająca wyrażeniu $\tilde{G}_i^{++}$(3) przedstawiona została niżej.

Tablica $\bar{Q}_i$:

q_i	adres q_i^k (	sygnał wyjściowy
q_1	x_7	y_2
q_2	x_1	y_3
q_3	x_4	y_1
q_4	x_{13}	y_3

Wyrażenie symboliczne $\tilde{G}_i^{++}$ (3) i tablica $\bar{Q}_i$ są głównymi elementami programowego wariantu automatu w pamięci komputera.

3.3. Programowy wariant automatu Moore'a i jego działanie

Jak już nadmieniono wyżej, wyrażenie symboliczne $\tilde{G}_i^{++}$ (3) i tablica $\tilde{Q}_i$ jednoznacznie reprezentują działanie rozpatrywanego automatu Moore'a. Należy jednak zwrócić uwagę na to, że zarówno $\tilde{G}_i^{++}$ jak i $\tilde{Q}_i$ są elamentami statycznymi w pamięci komputera i nie realizują działania automatu. Aby komputer (mikrokomputer) mógł swoim działaniem reprezentować działanie rozpatrywanego automatu Moore'a, w pamięci komputera powinien znajdować się odpowiedni program działający na wyrażeniu $\tilde{G}_i^{++}$ i tablicy $\tilde{Q}_i$. Oprócz tego programu, w pamięci komputera muszą być zarezerwowane następujące słowa:

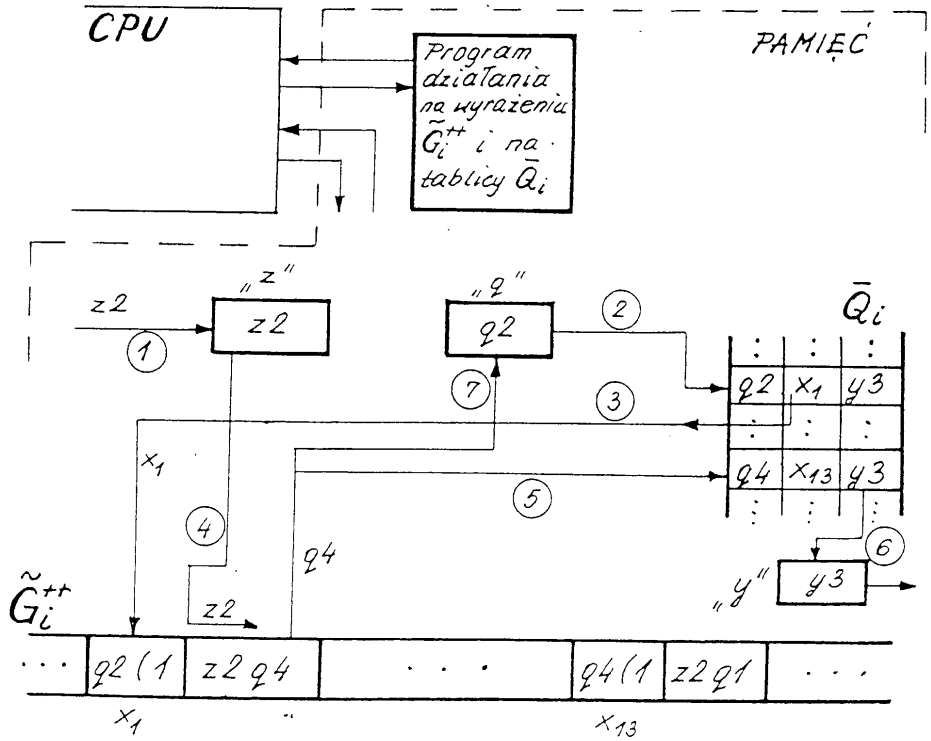
- słowo "z" do pamiętania aktualnego sygnału wejściowego $z_j \in Z$ automatu ;
- słowo "q" do pamiętania aktualnago stanu wewnętrznego $q_r \in Q$ automatu ;
- słowo "y" do pamiętania aktualnego sygnału wyjściowego $y_i \in Y$ automatu.

Konfiguracja elementów programowego wariantu automatu typu Moore'a przedstawiona została na rys.3.

Działanie programowego wariantu automatu łatwo wyjaśnić na podstawie pokazanej na rys.3 kolejności wykonywania operacji.

Aby zainicjować działanie automatu wywoływany jest program $\mathcal{A}$. Wykonywanie programu $\mathcal{A}$ przez CPU rozpoczyna się w chwili wprowadzenia z pulpitu jednego z symboli z_1, z_2 alfabetu wejściowego automatu. Załóżmy, że jest to symbol z_2. Symbol ten, reprezentujący sygnał wejściowy automatu, zapamiętywany jest w komórce "z". Operacja ta oznaczona jest na rys.3 symbolem (1) .W komórce "q" pamiętany jest stan początkowy q_2 automatu. W operacji oznaczonej symbolem (2) symbol q_2 jest szukany w tablicy $\tilde{Q}_i$. W wierszu q_2 tablicy $\tilde{Q}_i$ znajduje się adres x_1 wierzchołka q_2 w wyrażeniu $\tilde{G}_i^{++}$ pamiętanym w pamięci komputera. Na podstawie adresu x_1 odszukany jest w wyrażeniu $\tilde{G}_i^{++}$, w operacji (3) , symbol q2(1

reprezentujący wierzchołek q_2 grafu automatu. W komórkach znajdujących się za q2(1 znajdują się symbole krawędzi łączących wierzchołek q_2 z innymi wierzchołkami grafu. W operacji (4) komputer szuka w wymienionych komórkach symbolu z_2. Wynikiem tej operacji jest z2q4. Symbol q4 oznacza nowy stan automatu, dla którego powinien być wygenerowany odpowiedni sygnał wyjściowy y_i automatu zapisany w tablicy $\tilde{Q}_i$. Stąd też w operacji (5) w tablicy $\tilde{Q}_i$ szukany jest wiersz oznaczony symbolem q_4. W wierszu tym zapisany jest symbol y_3, który w operacji (6) pobierany jest z $\tilde{Q}_i$ do komórki "y". Sygnał ten wyświetlany jest na ekranie monitora lub inicjuje działanie programu wykonującego decyzję wygenerowaną przez automat. Po wygenerowaniu sygnału y_2 uaktualniany jest w operacji (7) nowy stan automatu, tj. stan q_4, który zapisywany jest w komórce "q". Operacje wykonywane w następnych taktach pracy automatu są identyczne z operacjami podanymi wyżej.



Rys.3. Elementy programowego wariantu automatu Moore'a w pamięci komputera ze wskazaniem na kolejność operacji

Przedstawiona wyżej idea programowej realizacji automatu skończonego może być odpowiednio modyfikowana, np. mogą być pominięte indeksy nawiasów.
W rozdziale czwartym niniejszej instrukcji scharakteryzowany zostanie wariant programowej realizacji automatu Moore'a praktycznie zrealizowany na mikrokomputerze IBM PC AT z pewnymi drobnymi modyfikacjami w stosunku do przedstawionej wyżej idei podstawowej.

3.4. Testowanie programowego wariantu automatu Moore'a o dużej liczbie stanów wewnętrznych

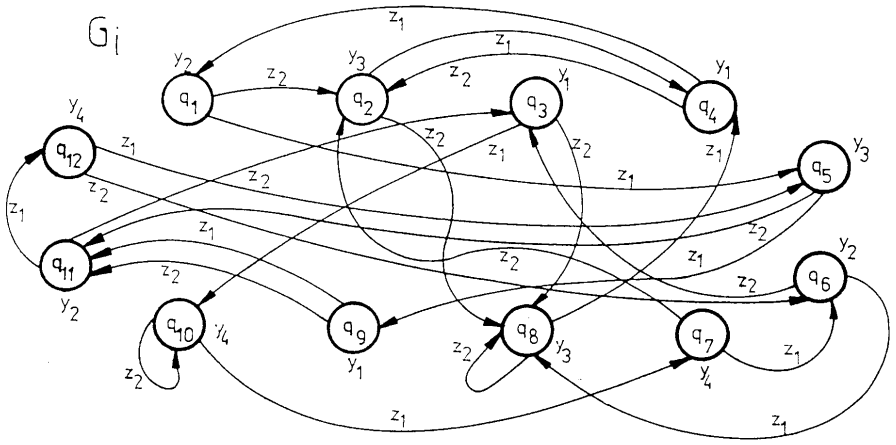
Większość automatów typu Moore'a, reprezentujących sobą procesy decyzyjne lub przetwarzanie określonej informacji dyskretnej, charekteryzuje się tym, że automaty te posiadają dużą liczbę stanów. Wyświetlanie na ekranie monitora kompletnego grafu przejść tego typu automatu, w celu sprawdzenia jego działania, jest wręcz niemożliwe. Stąd też zachodzi konieczność podziału, realizowanego przez komputer, grafu automatu na takie podgrafy, które mogą być wyświetlane na ekranie monitora. Kryterium podziału grafu automatu Moore'a o dużej liczbie stanów wewnętrznych wynika z konieczności przejścia z tego typu automatu na odpowiadający mu automat z parametrem wewnętrznym [6].
Zgodnie z tym kryterium graf automatiu dzieli się na podgrafy charakteryzujące się tym, że każdy z tych podgrafów może zawierać tylko wierzchołki q_r opisane różniącymi się symbolami $y_i \in Y$ i musi być pdgrafem spójnym. Oznacza to, że w danym podgrafie $G_{i,j}$ grafu G_i nie może być dwóch wierzchołków opisanych jednym i tym samym symbolem y_r. Jako przykład rozpatrzmy graf G_i, pewnego automatu $<A_i>$, przedstawiony na rys.4.

Wyrażenie symboliczne G_i^+ reprezentujące ten graf z pominięciem symboli y_r (r=1,2,...) przyporządkowanym wierzchołkom grafu ma następującą postać:

$$G_i^+ = {}^0(q_1{}^1(z_2q_2{}^2(z_1q_4{}^3(z_1q_1, z_2q_2)^3 z_2q_8{}^3(z_1q_4, z_2q_8)^3)^2 z_1q_5{}^2(z_1q_9{}^3($$

$$z_1 q_{11}{}^4 (z_1 q_{12}{}^5 (z_1 q_5, z_2 q_6{}^6 (z_1 q_8, z_2 q_3{}^7 (z_2 q_8, z_1 q_{10}{}^8 (z_1 q_7{}^9 ($$

$$z_1 q_6, z_2 q_2)^9, z_2 q_{10})^8)^7)^6)^5, z_2 q_3)^4, z_2 q_{11})^3, z_2 q_{11})^2)^1)^0 \qquad (4)$$



Rys.4. Przykładowy graf automatu Moore'a o dużej liczbie wierzchołków.

Wyrażenie G_i^+ jest pamiętane w liniowej sekwencji komórek pamięci. Aby uzyskać pełną reprezentację grafu G_i z rys.4, w pamięci komputera razem z grafem G_i, pamiętana jest tablica przyporządkowań stanom wewnętrznym q_r sygnałów wyjściowych y_n. Tablica ta ma postać jak niżej:

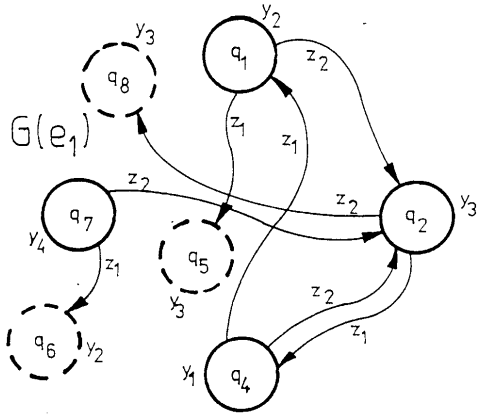
q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9	q_{10}	q_{11}	q_{12}
y_2	y_3	y_1	y_1	y_3	y_2	y_4	y_3	y_1	y_4	y_2	y_4

(5)

Na podstawie tablicy (5) możemy określić podzbiory $\tilde{Q}_{i,j}$ tych wierzchołków grafu G_i, których przyporządkowany jest jeden i ten sam symbol wyjściowy y_n.

$$
\begin{aligned}
y_1 &\longleftrightarrow \tilde{Q}_{i,1} = \{ q_3, q_4, q_9 \} \\
y_2 &\longleftrightarrow \tilde{Q}_{i,2} = \{ q_1, q_6, q_{11} \} \\
y_3 &\longleftrightarrow \tilde{Q}_{i,3} = \{ q_5, q_8, q_2 \} \\
y_4 &\longleftrightarrow \tilde{Q}_{i,4} = \{ q_7, q_{10}, q_{12} \}
\end{aligned} \qquad (6)
$$

Znając podzbiory $\tilde{Q}_{i,j}$, wyrażenie G_i^+ można podzielić na podstawie wyrażenia $G_{i,j}^+$ zawierające te elementy q_i stojące przed nawiasami otwierającymi $^k($, z których każdy przynależy do innego z podzbiorów $\tilde{Q}_{i,j}$. Każde z tych wyrazów reprezentuje odpowiedni podgraf $G_{i,j}$ grafu G_i z rys.4. i otrzymuje wartość e_r ($r = 1,2,..$) pewnego parametru e nazywanego parametrem wewnętrznym automatu.



Rys.5. Podgraf $G(e_1)$ grafu G_i z rysunku 4.

Pierwsze podwyrażenie $G_{i,1}^+$ wyrażenia $G_i^+(4)$ ma następującą postać:

$$
G_{i,1}^+ = {}^0(q_1{}^1(z_2q_2{}^2(z_1q_4{}^3(z_1q_1, z_2q_2)^3, z_2\boxed{q_8}{}^3(\ldots)^3)^2, z_1\boxed{q_5}{}^2(\ldots
$$

$$
\ldots q_7{}^9(z_1\boxed{q_6}, z_2q_2)^9, \ldots)^2)^1)^0 \longleftrightarrow e_1 \qquad (7)
$$

Podwyrażenie $G^+(e_1)$ wyrażenia $G^+(4)$ można sprowadzić do wyrażenia symbolicznego $G^{++}(e_1)$ o następującej postaci:

$$G^{++}_{i,1} = {}^{0}(q_1{}^{1}(z_2q_2{}^{2}(z_1q_4{}^{3}(z_1q_1, z_2q_2)^{3}, z_2\boxed{q_8})^{2}, z_1\boxed{q_5})^{1},$$

$$q_7{}^{1}(z_1\boxed{q_6}, z_2q_2)^{1})^{0} \longleftrightarrow e_1 \qquad (8)$$

Na podstawie wyrażenia $G^{++}_{i,1}$ można narysować podgraf $G_{i,1} = G(e_1)$ pokazany na rys.5. Symbole $\boxed{q_5}$, tj. symbole występujące w wyrażeniu $G^{+}_{i,1}$ w kwadratach, wskazują na takie wierzchołki podgrafu $G(e_1)$, które do niego nie należą. Wierzchołki te oznaczone zostały w podgrafie $G(e_1)$ linią przerywaną.

Drugie podwyrażenie $G^{+}_{i,2}$ wyprowadzone z wyrażenia $G^{+}_{i}(4)$ ma postać jak niżej:

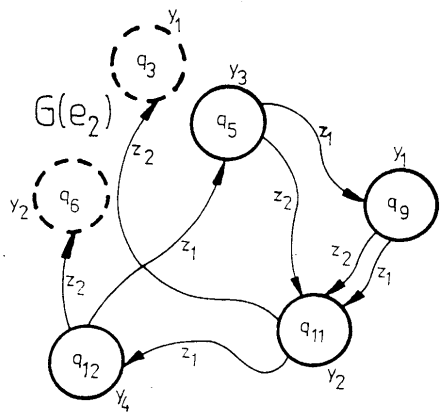
$$G^{++}_{i,2} = \ldots q_5{}^{2}(z_1q_9{}^{3}(z_1q_{11}{}^{4}(z_1q_{12}{}^{5}(z_1q_5, z_2\boxed{q_6}\ldots)^{5}, z_2\boxed{q_3})^{4},$$

$$z_2q_{11})^{3}, z_2q_{11})^{2}\ldots \longleftrightarrow e_2 \qquad (9)$$

Podwyrażenie $G^{+}_{i,2}$ można sprowadzić do wyrażenia $G^{++}_{i,2}$:

$$G^{+}_{i,2} = {}^{0}(q_5{}^{1}(z_1q_9{}^{2}(z_1q_{11}{}^{3}(z_1q_{12}{}^{4}(z_1q_5, z_2\boxed{q_6})^{4}, z_2\boxed{q_3})^{3},$$

$$z_2q_{11})^{2}, z_2q_{11})^{1})^{0} \longleftrightarrow e_2 \qquad (10)$$



Rys.6. Podgraf $G(e_2)$ grafu G_i z rys.4.

Na podstawie wyrażenia $G^{++}_{i,2}$ i przyporządkowań typu $q_r \longleftrightarrow y_n$ zawartych w tablicy (5) można narysować podgraf $G(e_2)$ grafuG_i z rys.4. Podgraf ten przedstawiony został na rys.6. Przedstawione na tym rysunku wierzchołki q_3 i q_6 oznaczone zostały linią przerywaną, ponieważ nie należą one do podgrafu $G(e_2)$.

Trzecie podwyrażenie $G^{+}_{i,3}$, i zarazem ostatnie, wyprowadzone z wyrażenia G^{+}_{i} ma następującą postać:

$$G^{+}_{i,3} = \ldots q_8{}^{3}(z_1\boxed{q_4},z_2q_8)^{3})^{2},\ldots q_6{}^{6}(z_1q_8,z_2q_3{}^{7}(z_2q_8,$$

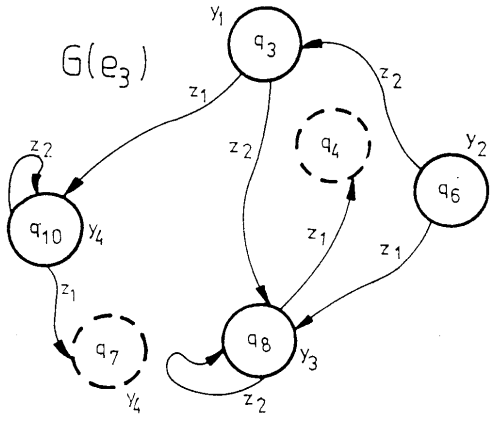
$$z_1q_{10}{}^{8}(z_1\boxed{q_7}{}^{9}(\ldots)^{9},z_2q_{10})^{8})^{7})^{6}\ldots \longleftrightarrow e_3 \qquad (11)$$

Podwyrażenie $G^{+}_{i,3}$ można przekształcić do postaci wyrażenia $G^{++}_{i,3}$:

$$G^{++}_{i,3} = {}^{0}(q_8{}^{1}(z_1\boxed{q_4},z_2q_8)^{1},q_6{}^{1}(z_1q_8,z_2q_3{}^{2}(z_2q_8,$$

$$z_1q_{10}{}^{3}(z_1\boxed{q_7},z_2q_{10})^{3})^{2})^{1})^{0} \longleftrightarrow e_3 \qquad (12)$$

Na podstawie tego wyrażenia można narysować podgraf $G(e_3)$ grafu G_i z rysunku 4. Podgraf $G(e_3)$ przedstawiony został na rys.7.



Rys.7.Pograf $G(e_3)$ grafu G_i z rysunku 4

Jak to już nadmieniano wyżej przy sprawdzaniu programowego wariantu o dużej liczbie stanów na ekranie monitora mogą być wyświetlane tylko fragmenty grafu automatu. Fragmentami tymi są zdefiniowane wyżej podgrafy $G_{i,j}$. Stąd też w programie sprawdzającym działanie zadanego programowego wariantu automatu $<A_i>$ znajduje się odpowiednia procedura, której realizacja powoduje podział wyrażeń typu G_i^+ (porównaj wyrażenie (4)) na podwyrażenia G_i^{++} zapamiętywane w odpowiednich obszarach pamięci. Z kolei inna procedura powoduje wyświetlanie na ekranie monitora odpowiedniego podgrafu $G_i(e_j)$ zależnego od aktualnie rozpatrywanego stanu q_r automatu. Procedura ta działa na podwyrażeniach $G_{i,j}^{++}$ i na tablicy T(e) zawierającej adresy termów $g_r^k($ występujących w podwyrażeniach $G_i^{++}(e_j)$. Początkowa część tablicy T(e), dotycząca wyrażeń $G^{++}(e_1)$,$G^{++}(e_2)$ i $G^{++}(e_3)$ przedstawina została na rys.8.

Symbol q_r	Adres $q_r^k($ w wyrażeniu $G_{i,j}$	Ardes początku wyrażenia $G_{i,j}^{++}$
q_1	x_{j_1}	$x(e_1)$
q_2	x_{j_2}	$x(e_1)$
q_3	x_{j_3}	$x(e_3)$
q_4	x_{j_4}	$x(e_1)$
q_5	x_{j_5}	$x(e_2)$
⋮	⋮	⋮

Rys.8. Struktura tablicy $T(e_j)$

Adresy zapisane w drugiej kolumnie tablicy T(e) są adresami symbolicznymi i ich rzeczywista wartość zależy od adresu $x(e_j)$ początku wyrażenia $G_{i,j}^{++}$, do którego dane q_r przynależy.

4. Opis programu umożliwiającego realizację automatu skończonego na mikrokomputerze

4.1. Ogólna koncepcja programu

W poprzednim punkcie przedstawiono ogólne zasady programowej realizacji automatów skończonych. Obecnie zostanie omówiony sposób implementacji programowego wariantu automatu typu Moore'a na mikrokomputerze IBM PC.
Z wcześniejszych rozważań wynika, że program reprezentujący swoim działaniem automat skończony powinien realizować trzy podstawowe funkcje:

- wprowadzanie danych określających strukturę automatu i wstępne sprawdzenie ich poprawności,
- symulację działania automatu (program $\mathcal{A}$ opisany w pkt. 1.3),
- graficzną prezentację działania automatu na ekranie mikrokomputera.

Poniżej zostanie przedstawiony sposób realizacji tych funkcji z zaznaczeniem różnic w stosunku do wcześniej omówionych założeń.

4.1.1. Wprowadzanie struktury automatu do pamięci komputera

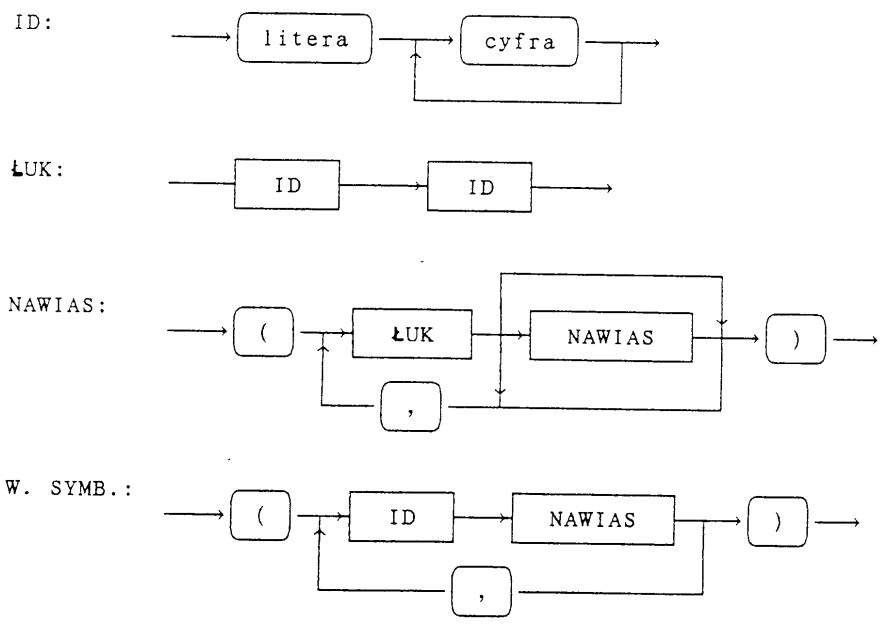
Struktura automatu skończonego reprezentowana jest w pamięci komputera w postaci wyrażenia symbolicznego. W omawianej implementacji wyrażenie symboliczne może być wprowadzane zarówno w wersji G_i^+ (1) lub wersji G_i^{++} (2). Nie podaje się natomiast indeksów nawiasów (są one automatycznie dołączane przez procedurę czytającą dane wejściowe). Oprócz wyrażenia symbolicznego podaje się także zbiór symboli wejściowych Z, zbiór stanów wewnętrznych Q, zbiór symboli wyjściowych Y oraz funkcję wyjść reprezentowaną przez tablicę $\bar{Q}_i(q_r \longleftrightarrow y_j)$.

Po wprowadzeniu wyrażenia symbolicznego poddawane jest ono analizie (sprawdzana jest poprawność jego składni). W tym celu wykorzystywane są diagramy składni wyrażenia symbolicznego przed-

stawione na rys. 9. Zastosowany algorytm analizy składniowej charakteryzują dwie cechy:

- określenie kolejnego kroku analizy zależy tylko od obecnego stanu oraz od pojedyńczego, aktualnie wczytywanego symbolu,
- żadnego z kolejnych kroków analizy nie można cofnąć.

Jest to tzw. analiza bez powrotów z wyprzedzeniem o jeden symbol (z jęz. ang. one-symbol-lookahead without backtracking) [9].



Rys. 9. Diagramy składni wyrażenia symbolicznego

W przypadku stwierdzenia poprawności wyrażenia symbolicznego umieszczane jest ono w pamięci, w przeciwnym przypadku sygnalizowany jest błąd. Ostatecznie w pamięci pamiętane jest wyrażenie symboliczne $\tilde{G}_i^{++}$ oraz tablica $\tilde{Q}_i$.

Wraz z analizą składni wyrażenia symbolicznego tworzona jest lista łuków grafu automatu. Jest ona wykorzystywana do graficznej prezentacji grafu automatu na ekranie komputera i dokładniej zostanie opisana później.

4.1.2. Symulacja działania automatu

Procedura symulująca działanie automatu skończonego oparta jest na algorytmie przedstawionym w pkt. 1.3. Została ona rozszerzona o podprogram umożliwiający wprowadzanie sekwencji symboli wejściowych z pulpitu komputera. Po wczytaniu symbolu wejściowego wykonywane są kolejne operacje zgodnie ze schematem przedstawio-/nym na rys. 3. W ich wyniku otrzymujemy nowy stan automatu q_r oraz odpowiadający mu symbol wyjściowy y_j. W trakcie symulacji działania automatu na ekranie wyświetlany jest graf automatu. Wierzchołek grafu odpowiadający aktualnemu stanowi automatu jest odpowiednio oznaczony. Wraz z analizą kolejnych symboli wejściowych następuje uaktualnienie obrazu grafu na ekranie komputera. Sposób uzyskania grafu automatu na ekranie komputera zostanie przedstawiony w następnym punkcie.

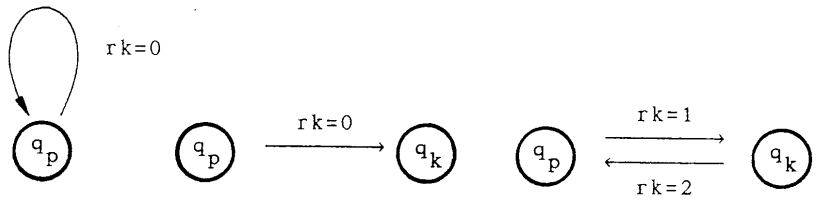
4.1.3. Graficzne przedstawienie grafu automatu na ekranie komputera

Jednym ze sposobów opisu działania automatu skończonego jest podanie jego grafu przejść. Graf automatu reprezentowany jest wówczas przez zbiór wierzchołków V odpowiadający stanom wewnętrznym automatu oraz przez zbiór krawędzi E określający możliwe przejścia między stanami automatu. W celu uzyskania na ekranie odpowiadającego badanemu automatowi grafu przejść należy określić na podstawie wyrażenia symbolicznego zbiór krawędzi tego grafu. Krawędź opisywana jest następującymi parametrami:

$$k = (q_p, q_k, z_j, rk) \text{ , gdzie}$$

q_p - stan początkowy (aktualny),
q_k - stan końcowy (następny),
z_j - sygnał wejściowy powodujący przejście automatu ze q_p do stanu q_k,
rk - rodzaj krawędzi.

Lista krawędzi tworzona jest równolegle podczas analizy wyrażenia symbolicznego (patrz pkt. 4.1.1). Otrzymujemy wówczas zbiór krawędzi dla dla których okreslone są q_p,q_k oraz z_j. Następnie zostaje określony rodzaj każdej krawędzi zgodnie z oznaczeniami przedstawionymi na rys. 10.



Rys.10. Rodzaje krawędzi występujące w grafie automatu

Dysponując opisanym zbiorem krawędzi można narysować graf automatu. Odpowiednia procedura składa się z trzech części:
- rysowanie wierzchołków (stan aktualny jest wyróżniony),
- rysowanie łuków (krawędzi) odpowiadających przejściom między stanami,
- opisanie krawędzi grafu odpowiednimi symbolami wejściowymi z_j.

Rozmiar grafu jaki można uzyskać na ekranie jest ograniczony. Dlatego też istnieje możliwość częściowego rysowania grafu (tzn. rysowania tylko istotnych w danym momencie fragmentów).

4.2. Opis działania programu

Menu główne programu umożliwia wybranie jednej z następujących opcji:
- DANE
- EDYCJA
- SYMULACJA

Wybór żądanej opcji następuje w wyniku wprowadzenia z klawiatury wyróżnionego znaku dla odpowiedniego pola menu lub za pomocą klawiszy kierunkowych (strzałek) i klawisza >ENTER<. Rezygnacja z wybranej opcji następuje w wyniku naciśnięcia klawisza >ESC<.

4.2.1. Opcja DANE

Wybór tej opcji umożliwia realizację jednej z następujących funkcji:

- wczytanie danych z dysku,
- zapisanie danych na dysk,
- inicjowanie wprowadzania danych,
- przeglądanie katalogu,
- zmiana katalogu,
- wyjście z programu.

Do danych określających strukturę automatu należą:

- zbiór symboli wejściowych Z,
- zbiór stanów wewnętrznych Q,
- zbiór symboli wyjściowych Y,
- funkcja przejść (w postaci wyrażenia symbolicznego),
- funkcja wyjść (w postaci tablicy $\overline{Q}_i$).

Po wczytaniu danych są one wyświetlane na ekranie komputera wraz z odpowiadającym im grafem automatu.

4.2.2. Opcja EDYCJA

Opcja ta umożliwia wprowadzanie lub modyfikację danych określających strukturę automatu. Przy wprowadzaniu danych należy pamiętać, że:

- identyfikator określający symbol wejściowy, stan wewnętrzny lub symbol wyjściowy może składać się z poje yńczej litery i odpowiedniego indeksu np z_1, q_5, y_3,
- poszczególne elementy wprowadzanych danych oddzielane są przecinkiem np $Z=\{z_1, z_2, z_3\}$,
- przy wprowadzaniu wyrażenia symbolicznego nie podaje się indeksów nawiasów (są one dodawane automatycznie przez program).

W przypadku wprowadzenia błędnych danych sygnalizowany jest rodzaj błędu oraz miejsce jego wystąpienia. Przy poprawnym zakończeniu edycji na ekranie zostają wyświetlone aktualne dane oraz odpowiadający im graf automatu.

4.2.3. Opcja SYMULACJA

Opcja SYMULACJA umożliwia testowanie automatu, którego struktura została wczytana ze zbioru dyskowego lub wprowadzona za pomocą opcji EDYCJA. Po jej wyborze istnieje możliwość realizacji jednej z następujących funkcji:

- zmiana aktualnago stanu wewnętrznego automatu,
- wprowadzenie słowa wejściowego (ciągu symboli wejściowych),
- praca ciągła automatu,
- praca krokowa automatu.

Po określeniu słowa wejściowego można testować automat na dwa sposoby:

- automat generuje na podstawie słowa wejściowego słowo wyjściowe (pracy automatu nie można wówczas przerwać),
- praca automatu śledzona jest krok po kroku (po każdym symbolu wejściowym praca automatu jest wstrzymywana).

Na ekranie można obserwować kolejne zmiany stanu automatu, a ponadto wyświetlana jest dokumentacja jego działania zawierająca wydruk kolejnych stanów wewnętrznych oraz generowanych symboli wyjściowych.

W trakcie testowania automatu może powstać sytuacja, że dane przejście w automacie jest nieokreślone (w przypadku automatów niezupełnych). Następuje wówczas sygnalizacja błędu oraz wstrzymywana jest dalsza praca automatu.

Podczas testowania automatu istnieje również możliwość zmiany jego aktualnego stanu bez wprowadzania symbolu wejściowego.

Przebieg ćwiczenia

Każdy student otrzymuje graf automatu typu Moore'a i na jego podstawie buduje wyrażenie symboliczne G_i^+, reprezentujące ten graf oraz tablicę $\overline{Q}_i(q_i \longleftrightarrow y_r)$. Wyrażenie G_i^+ przekształcane jest do postaci dogodnej do zapamiętania w pamięci komputera. W następnej kolejności przekształcone wyrażenie G_i^+, jak również tablica $\bar{Q}_i$ wprowadzane są z pulpitu do pamięci komputera.

Po wprowadzeniu G_i^+ i $\overline{Q}_i$ do pamięci uruchomiany jest program działający na tych elementach. W wyniku uruchomienia tego programu na ekranie wyświetlany jest pierwszy wierzchołek grafu G_i reprezentujący stan początkowy automatu $<A_i>$. Następnie student wprowadza z pulpitu komputera sygnały wejściowe $z_j \in Z$. Pod wpływem tych sygnałów na ekranie wyświetlane są kolejne fragmenty grafu automatu. Dla każdego ciągu wejściowego $z_{j1}, z_{j2}..z_{jn}$ należy odrysować z ekranu odpowiadający mu fragment grafu automatu. Podgrafy te należy umieścić w sprawozdaniu.W następnej kolejności należy wprowadzić z pulpitu taki minimalny ciąg symboli $z_{j1}, z_{j2}..z_{jn}$ pod wpływem którego pojawiłby się na ekranie kompletny graf automatu.

Następna część ćwiczenia obejmuje sprawdzenie działania programowego wariantu automatu Moore'a o dużej liczbie stanów wewnętrznych. W tej części ćwiczenia studenci otrzymują od prowadzącego laboratorium wyrażenie symboliczne i tablicę wyjść reprezentującą graf automatu o dużej liczbie stanów wewnętrznych. Wyrażenie to, razem z tablicą wyjść automatu, wprowadzane jest z pulpitu do pamięci mikrokomputera. Następnie uruchomiony jest program symulujący działanie automatu. Program ten nie wyświetla podgrafów automatu a tylko ciągi symboli y_i i stanów wewnętrznych q_r zależnych od podanego ciągu symboli wejściowych z_i. Studenci sprawdzają na podstawie tych ciągów czy działanie automatu jest zgodne z otrzymanym wyrażeniem symbolicznym i tablicą wyjść.

W następnej kolejności uruchamiany jest program sprawdzania działania automatu przy jednoczesnym wyświetlaniu pografów grafu badanego automatu. Studenci testują automat odpowiednią sekwencją sygnałów wejściowych, aby uzyskać wszystkie możliwe podgrafy i przejścia między nimi. Wyświetlone podgrafy należy odrysować a następnie na ich podstawie należy określić podwyrażenie $G_{i.j}^+$ zadanego wyrażenia G_i^+.

Sprawozdanie z ćwiczenia

W sprawozdaniu należy:

- umieścić temat i cel ćwiczenia,
- umieścić graf automatu podany przez prowadzącego ćwiczenie,
- napisać wyrażenie symboliczne reprezentujące ten graf i przekształcić je na postać dogodną do zapamiętania w pamięci komputera,
- umieścić wyniki testowania programowego wariantu automatu o grafie zadanym przez prowadzącego,
- podać krótkie wnioski z ćwiczenia.

Literatura

1. Bromirski J., Teoria automatów, WNT, Warszawa 1972
2. Hopcroft J.E., Ullman J.D., Introduction to Automata Theory, Languages and Computation, Addison-Wesley Publishing Company, Reading, Massachusets-1979
3. Kazimierczak J., System cybernetyczny,Wyd. "Wiedza Powszechna" (seria "Omega"), Warszawa 1978
4. Kazimierczak J., Elementy syntezy formalnej systemów operacyjnych, Biul. WASK, Wyd. Politechniki Wrocł., Wrocław 1979
5. Kazimierczak J., Kluska J., Kaczmarek A., Podstawy teorii automatów - Laboratorium, Wyd. Politechniki Rzeszowskiej, Rzeszów 1984
6. Kazimierczak J., Łysakowska B., Intalligent adaptive control of a mobile robot: The automaton witha internal and extarnal parameter approach, "Robotica" (1989),volume 6,pp.319-326.
7. Klingman E.E., Projektowanie systemów mikroprocesorowych, WNT, Warszawa 1982
8. Kowalski St., Mostowski A.W., Teoria automatów i lingwistyka matematyczna, PWN, Warszawa 1979
9. Wawiłow E.N., Portnoj G.P., Synteza układów elektronicznych maszyn cyfrowych, WNT, Warszawa 1967
10. Wirth N., Algorytmy + struktury danych = programy, WNT, Warszawa 1989