ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1978

Seria: AUTOMATYKA z. 42 Nr kol. 568

Jerzy IHNATOWICZ

ZASTOSOWANIE MODELU MASZYNY O SKOŃCZONEJ LICZBIE STANÓW DO PROJEKTOWANIA ALGORYTMU STEROWANIA PROCESEM TECHNOLOGICZNYM

Streszczenie. Praca zawiera rozważania dotyczące metod sterowania maszyną o skończonej liczbie stanów (maszyną M) w taki sposób, by mając dane dotyczące stanu początkowego osiągnąć stan końcowy wykorzystując pewne kryterium optymalizacji. Przedstawiono przykład programu na maszynę cyfrową, który znajduje ciąg instrukcji maszyny M zapewniający dojście do zadanego stanu końcowego przy nieznajomości ogólnej liczby stanów i możliwości przechodzenia ze stanu poprzedniego do następnego. Umożliwia to formułowanie zadania w postaci programu rozmytego.

1. Wstęp

Zagadnienie sterowania maszyną o skończonej liczbie stanów zostało omówione przez R. Jakubowskiego i A. Kasprzaka w [1]. Autorzy przedstawili, opierając się na pracy S.H. Changa [3], możliwość zbudowania algorytmu wykonywania elementarnego programu rozmytego, w skład którego wchodzą trzy zasadnicze elementy:

- maszyna o skończonej liczbie stanów,
- program rozmyty, którego zadaniem jest sterowanie maszyną o skończonej liczbie stanów,
- zależność określająca związek między sterowaną maszyną i programem rozmytym.

Maszyna o skończonej ilości stanów zdefiniowana jest jako system:

$$M = \langle K, X, \Psi, X_0, T \rangle$$

gdzie:

K - skończony, niepusty zbiór instrukcji maszyny (np. zbiór czynności, które może wykonać obrabiarka),

X - skończony, niepusty zbiór stanów maszyny (np. zbiór wszystkich możliwych konturów, które można uzyskać wykorzystując w dowolny sposób zbiór K),

$X_0 \in X$ - początkowy (wyjściowy) stan maszyny, bądź zbiór początkowych stanów maszyny (np. kontur półfabrykatu, bądź zbiór konturów półfabrykatów),

$T \in X$ - końcowy stan maszyny, bądź zbiór końcowych stanów maszyny (np. kontur wyrobu finalnego, bądź zbiór konturów wyrobów finalnych),

Ψ - funkcja określająca sposób zmiany stanu maszyny ze stanu $X_i \rightarrow X_j$, gdzie $X_i, X_j \in X$.

Funkcję tę określa się następująco:

$$\bigwedge_{\substack{X_i \in X \\ \mu \in K}} \bigvee_{X_j \in X} \left\{ X_j = \Psi(X_i, \mu) \right\}$$

Sterowanie zdefiniowaną w ten sposób maszyną o skończonej liczbie stanów polega zatem na takim wykorzystaniu instrukcji μ ze zbioru K, by wychodząc ze stanu X_o, poprzez pewne stany pośrednie $X_i \in X$ osiągnąć stan końcowy T.

<u>Program maszyny</u> zdefiniowany jest [1] jako ciąg instrukcji:

$$\bar{\mu} = \mu_1, \mu_2, \ldots, \mu_n \in K$$

który zapewnia, że:

$$\Psi(X_o, \bar{\mu}) = \Psi(\ldots\Psi(\Psi(X_o, \mu_1), \mu_2), \ldots, \mu_n) \wedge T \neq \emptyset$$

W tym więc ujęciu, sterowanie maszyną o skończonej liczbie stanów jest wykonaniem programu maszyny.

Automatyzacja projektowania ciągu operacji technologicznych polega zatem na zbudowaniu odpowiedniego modelu maszyny o skończonej liczbie stanów, opisującego możliwie dokładnie dany proces, a następnie na zbudowaniu algorytmu znajdowania takiego ciągu $\bar{\mu}$, który spełnia warunki podane powyżej.

Każdy proces charakteryzują pewne ograniczenia i sekwencje wykonywanych czynności, które są z reguły opisywane przez technologa w bardzo ogólny i nieprecyzyjny sposób. Stąd też wynikła koncepcja zastosowania programu rozmytego do automatyzacji projektowania ciągu operacji technologicznych [1].

<u>Program rozmyty</u> jest ciągiem instrukcji rozmytych:

$$\bar{a} = a_1, a_2, \ldots, a_n \in \Sigma$$

gdzie Σ jest niepustym zbiorem instrukcji rozmytych.

Każda z tych instrukcji to wspomniane powyżej nieprecyzyjne informacje podawane przez technologa.

Odpowiednia interpretacja programu rozmytego $\bar{a}$ powinna w wyniku doprowadzić do otrzymania ciągu instrukcji $\bar{\mu}$. Istotą zagadnienia staje się więc znalezienie takiego algorytmu, który "tłumaczy" ciąg $\bar{a}$ na ciąg $\bar{\mu}$.

Maszyna rozmyta została zdefiniowana jako [1]:

$$\hat{M} = \langle \Sigma, M, f, \lambda \rangle$$

gdzie:

Σ - niepusty zbiór instrukcji rozmytych,
M - maszyna o skończonej liczbie stanów,
f - funkcja możliwości $f: X \times \Sigma \times K \rightarrow \{0,1\}$,
λ - funkcja wykonywania $\lambda: X \times \Sigma \times K \rightarrow [0,1]$.

Funkcja f sprawdza wykonywalność danej instrukcji rozmytej przez maszynę M. Jeśli bowiem:

- maszyna M jest w stanie $X_i \in X$,
- została użyta taka instrukcja rozmyta $a \in \Sigma$, że można ją zinterpretować jako pewną instrukcję $\mu \in K$, taką, że:

$$\varphi(X_i, \mu) = X_j \in X$$

wówczas $f(X_i, a, \mu) = 1$ i instrukcja μ jest wykonaniem (interpretacją) a. W innym przypadku $f(X_i, a, \mu) = 0$.

Funkcja λ stanowi kryterium, według którego z kilku możliwych do zastosowania instrukcji μ (które zostały określone funkcją f) wybiera się jedną instrukcję μ.

Funkcja λ może przykładowo określać koszt zastosowania danej instrukcji μ. Jeśli zatem

$$\lambda(X_i, a, \mu_i) < \lambda(X_i, a, \mu_j),$$

a celem byłaby minimalizacja kosztów, wówczas należałoby zastosować instrukcję μ_i.

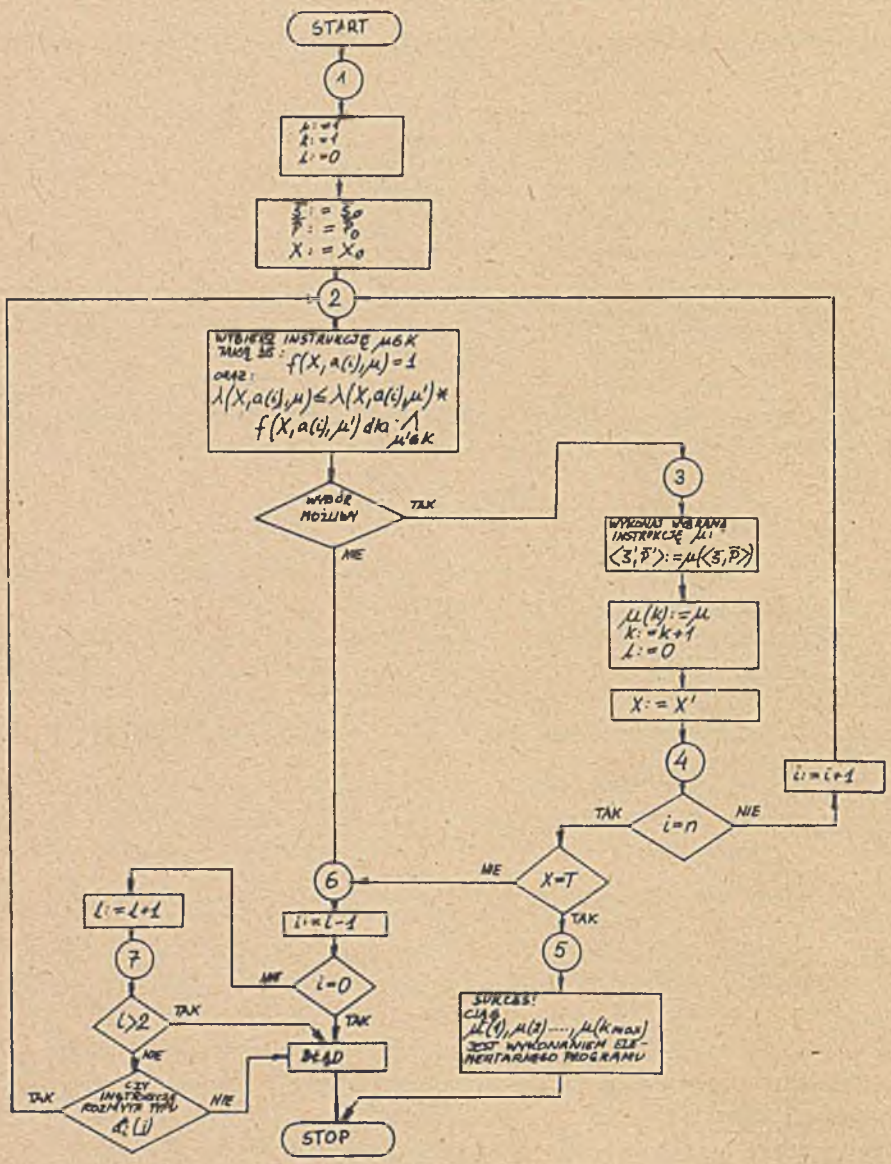
Mając więc określoną funkcję f i λ, można przyporządkować danej instrukcji rozmytej a instrukcję μ będącą najtańszą z możliwych do zastosowania.

W pracy [1] zaprezentowano koncepcję zastosowania programu rozmytego do projektowania ciągu technologicznego procesu obróbki skrawaniem. Przyjęto, że każdy ze stanów maszyny M (maszyny o skończonej liczbie stanów) jest opisem konturu osiowego przekroju poprzecznego obrabianego elementu.

Kontur ten jest opisany za pomocą dwóch ciągów parametrów, czyli:

$$X_i = \langle \bar{a}_i, \bar{P}_i \rangle \qquad X_i \in X$$

Ciągi $\bar{a}_i$ stanowią opis kształtu konturu, zaś ciągi $\bar{P}_i$ określają dodatkowe parametry technologiczne.



Rys. 1. Algorytm wykonywania elementarnego programu rozmytego wg Jakubowskiego i Kasprzaka [1]

Dokładne definicje zostały przedstawione w pracy [2] dotyczącej opisu i rozpoznawania kształtów elementów maszyn.

Ogólny schemat blokowy programu na maszynę cyfrową, zawierającego algorytm wykonywania elementarnego programu rozmytego zbudowany w myśl założeń przedstawionych powyżej - ilustruje rys. 1.

Schemat ten zaprezentowany w [1] pokazuje, w jaki sposób mając dane funkcje f i λ, opis instrukcji μ maszyny M - można wykonać elementarny program rozmyty

$$\bar{a} = a_i(1),\ a_i(2),\ldots,a_i(n)_{i=1,1,\ldots,m}$$

gdzie:

i - numer instrukcji rozmytej.

W programie tym dopuszcza się występowanie instrukcji rozmytych typu:

$$a_i^*(j)_{i=1,2,\ldots,m}$$

określających wielokrotne powtórzenie instrukcji rozmytej a_i w j-tym kroku elementarnego programu rozmytego, przy czym ilość powtórzeń jest nieokreślona, wiadomo jedynie, że program rozmyty składa się z n kroków.

Liczba powrotów do poprzedniej instrukcji rozmytej, określona przez l może być co najwyżej równa 2, umożliwia to dokonanie ponownego wyboru instrukcji μ, o ile wystąpiła w programie instrukcja typu $a_i^*(j)$.

2. Sformułowanie problemu

Zastosowanie algorytmu podanego przez R. Jakubowskiego i A. Kasprzaka (rys. 1) wymaga znajomości funkcji f (funkcja możliwości) i λ (funkcja wykonywania).

Określenie f i λ może być problemem dość złożonym, w przypadku gdy zbiory X oraz K zawierają wiele elementów. Znajomość funkcji f jest zaś konieczna, gdyż funkcja f opisuje, które przejścia ze stanu $X_i \to X_j / X_i, X_j \in X$ są możliwe i które instrukcje μ należy przy tym stosować.

Analizowanie maszyny o skończonej liczbie stanów jest zagadnieniem bardzo uciążliwym, celowe więc stało się zbudowanie takiego algorytmu wykonywania elementarnego programu rozmytego, którego użycie nie wymaga znajomości funkcji możliwości.

W niniejszym artykule będzie przedstawiony taki algorytm, zaś w dalszej jego części przykład jego zastosowania. Algorytm ten, podobnie jak

przedstawiony na rys. 1, pozwala znaleźć najtańszy z możliwych ciągów $\bar{\mu}$ zapewniających osiągnięcie końcowego stanu T maszyny M.

Dopuszcza on ponadto jeszcze bardziej ogólny opis zadania - taki, który składa się z jednej tylko instrukcji rozmytej:

"osiągnąć stan T, wychodząc ze stanu X_0, korzystając w taki sposób ze zbioru K, by minimalizować koszty na każdym etapie".

Nie wymaga więc podawania nawet ciągu $\bar{a}$.

3. Opis algorytmu

Schemat blokowy omawianego algorytmu umieszczono na rys. 2. Jak wspomniano wyżej, stosując ten algorytm znajduje się taki ciąg instrukcji $\bar{\mu}$, który (o ile istnieje) pozwala osiągnąć stan końcowy T, wychodząc ze stanu X_0, stosując kolejne instrukcje ciągu $\bar{\mu}$.

Przy przechodzeniu z każdego poprzedniego do następnego stanu maszyny M wybierana jest zawsze ta instrukcja μ ze zbioru K, która charakteryzuje się najmniejszym kosztem.

Przyjmując, że:

- i - oznacza numer kolejnego stanu maszyny M,
- X_i - jest opisem i-tego stanu maszyny M,
- $\Psi(X_i,\mu)$ - efekt wykonania instrukcji w stanie X_i, przy czym zachodzi, że:

$$\bigwedge_{i,j\in N} \bigwedge_{\mu\in K} \left\{ X_j = \Psi(X_i,\mu) \in X \right\}$$

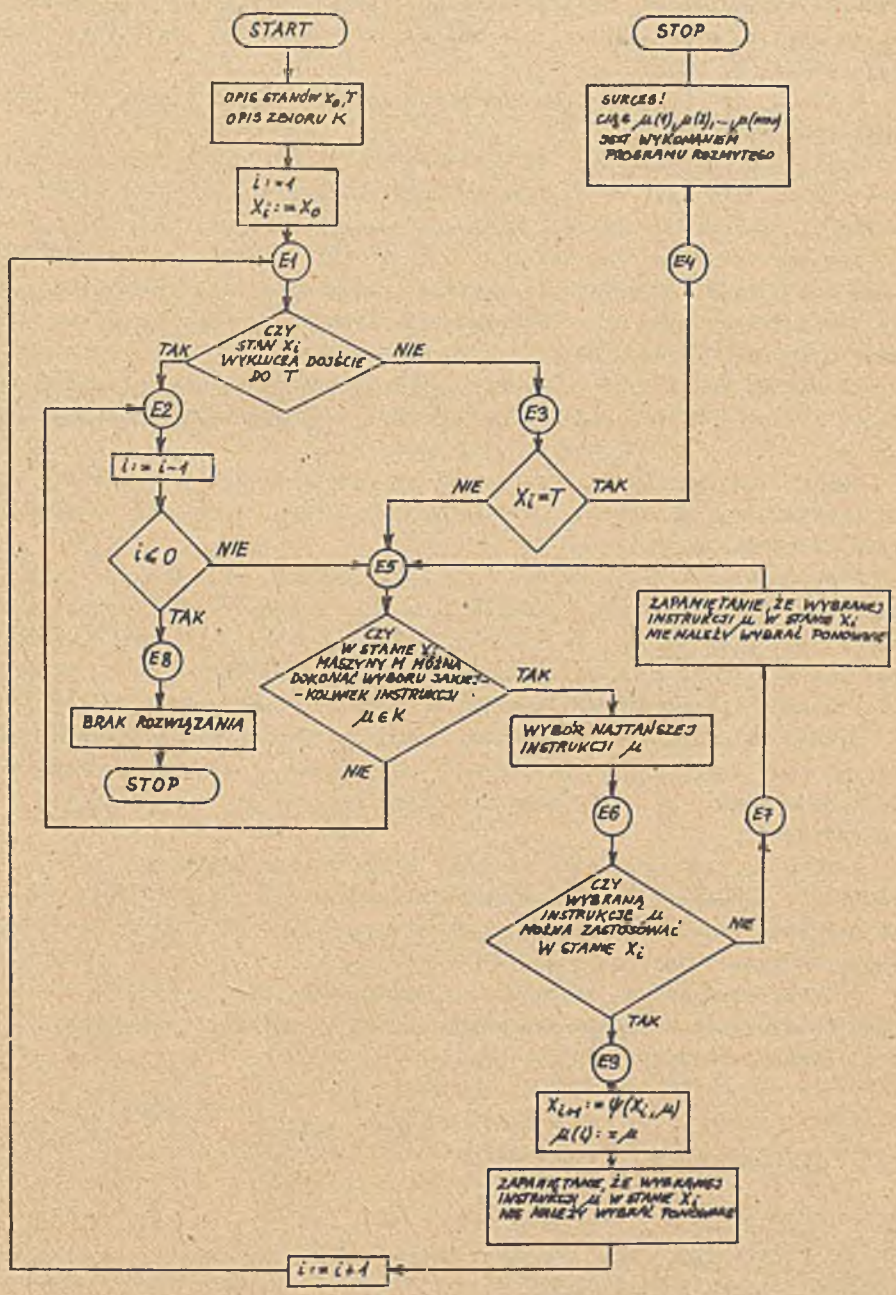
działanie algorytmu można przedstawić następująco:

Po wprowadzeniu danych, którymi są opisy stanów X_0 i T maszyny M oraz opis instrukcji $\mu \in K$, za pierwszy stan maszyny M zostaje przyjęty stan wyjściowy X_0.

W dalszym ciągu jest przeprowadzone badanie, czy aktualny stan maszyny jest taki, że wyklucza dojście (możliwość dojścia) do stanu końcowego T. (W tym kroku następuje więc sprawdzenie takiego kryterium określonego przy opisie danego procesu technologicznego, że niespełnienie go uniemożliwia prowadzenie procesu w poprawny sposób).

Jeśli zostanie wykryte, że ze stanu badanego X_i nie będzie można przejść do stanu T, następuje wykonanie kroku E2, w przeciwnym zaś razie kroku E3.

W kroku E3 bada się, czy aktualny stan X_i jest już stanem końcowym. T. Jeśli nie jest, tzn. $X_i \neq T$, zostanie wykonany krok E5, w którym określa się, czy w stanie X_i można użyć którejkolwiek z instrukcji μ.



Rys. 2. Algorytm wykonywania elementarnego programu rozmytego nie wymagający podawania funkcji możliwości f

Jeśli jest to możliwe, wybrana zostaje najtańsza instrukcja z możliwych do zastosowania, po czym wykonany zostaje krok E6. W przypadku, gdy okaże się, że w stanie X_i nie można już użyć żadnej instrukcji - następuje powrót do E2.

W kroku E6 zostaje wykonana próba zastosowania w stanie X_i wybranej instrukcji μ.

Jeśli zostanie wykryte, że wybranej instrukcji nie można użyć (no. zachodzi, że $\Psi(X_i,\mu) = X_i$), fakt ten zostaje zapamiętany w kroku E7,po czym następuje powrót do E5. W przeciwnym razie wykonywany jest krok E9. W kroku tym zostaje określony następny - X_{i+1} stan maszyny M, osiągnięty w wyniku zastosowania wybranej instrukcji μ w stanie X_i, zapamiętana zostaje użyta instrukcja oraz fakt, że w tym stanie (X_i) nie można jej już ponownie wybrać, po czym następuje powrót do E1.

Wykonanie kroku E2 zapewnia możliwość cofnięcia się do poprzedniego stanu maszyny M i dokonania wyboru innej niż poprzednio wybranej instrukcji μ.

Dopóki nie zostaną zbadane wszystkie inne kombinacje instrukcji tworzące szukany ciąg $\bar{\mu}$ maszyny M - można się cofać i dokonywać ponownych wyborów instrukcji μ.

Procedura taka jest powtarzana dopóty, dopóki nie zostanie osiągnięty stan T - nastąpi wówczas przejście do E4 i zakończenie działania algorytmu, zaś ciąg $\mu(1),\mu(2),\mu(3),\ldots,\mu(i_{max})$ jest szukanym ciągiem,bądź też zostaną zbadane wszystkie możliwe kombinacje - oznaczać to będzie,że rozwiązanie nie istnieje i działanie algorytmu zostanie zatrzymane.

Użycie opisanego algorytmu nie wymaga znajomości ilości stanów maszyny M, nie trzeba przeto analizować, czy z danego stanu X_i można przejść do innego stanu X_j, przy pomocy których instrukcji i przez jakie stany pośrednie.

Liczba wszystkich stanów maszyny M pozostaje nieznana,nie trzeba jej badać ani określać, analizowane są bowiem tylko te stany, które nie wykluczają możliwości uzyskania rozwiązania, przy czym badanie to jest wykonywane samoczynnie.

Przykład ilustrujący sposób przeprowadzenia takiego badania oraz możliwości zastosowania zaprezentowanego algorytmu zostanie przedstawiony w dalszej części niniejszej pracy.

4. Przykład

4.1. Opis zadania

Dane jest następujące zadanie, postawione przez technologa, omawiające proces obróbki skrawaniem:

"Zbadać, czy używając danego zestawu czynności obrabiarki, można zaprojektować taki ciąg operacji technologicznych, który pozwoli na

wykonanie pewnego elementu, którego kształt jest dany przez opis konturu osiowego przekroju poprzecznego.
Opis kształtu półfabrykatu, z którego ma być wykonany element, jest podany w ten sam sposób.
Jeśli zadanie to da się rozwiązać, zaprojektować taki ciąg, uwzględniając ponadto warunek, że należy wybierać zawsze najtańszą z możliwych do zastosowania czynności wykonywanych przez obrabiarkę".

4.2. Opis kształtów konturów elementów

Przyjmijmy, że opisy kształtów konturów elementów (półfabrykatów oraz wykonywanego elementu) są podane w postaci dwóch ciągów liczbowych, z których pierwszy opisuje ogólny kształt konturu, drugi zaś podaje wymiary.

Opis półfabrykatu będzie stanem X_o maszyny o skończonej liczbie stanów, opis wykonywanego elementu będzie odpowiednio stanem T.

Stany te będą więc scharakteryzowane następująco:

$$X_o = \langle XO, XWO \rangle$$

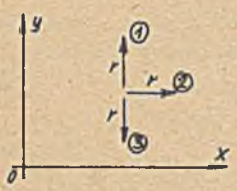
$$T = \langle T, TW \rangle$$

gdzie:

XO, T - ciągi opisujące ogólny kształt konturu (np. kierunku),
XWO, TW - ciągi zawierające informacje o wymiarach.

Rzecz jasna, każdy inny stan maszyny o skończonej liczbie stanów opisuje się analogicznie:

$$X_i = \langle XO_i, XWO_i \rangle$$



Rys. 3. Interpretacja graficzna kierunków użytych do opisu konturu w omawianym przykładzie

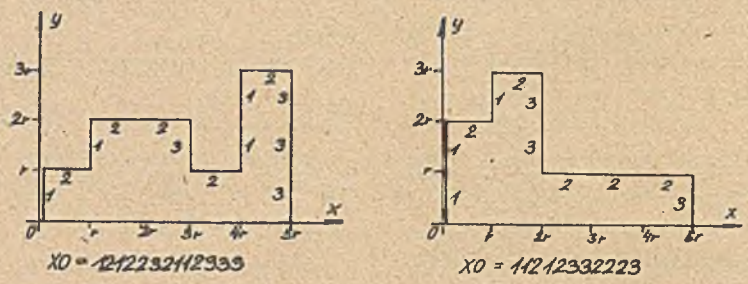
W przykładzie ograniczono się do konturów, które można opisać odcinkami prostymi, biegnącymi w 3 kierunkach. Ilustruje to rys. 3.

Każdy z odcinków ma jednostkową długość r.

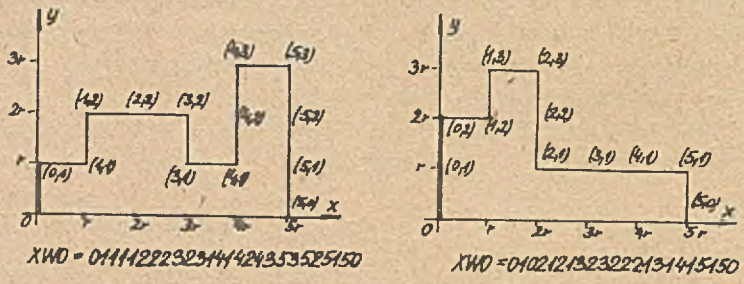
Przykładowe opisy kształtów konturów dwóch różnych osiowych przekrojów poprzecznych obrabianych elementów ilustruje rys. 4. (Metoda opisu ogólnego kształtu konturu (tworzenia XO) została zaczerpnięta z [2], zmodyfikowano ją jednak dla potrzeb omawianego przykładu).

Ciągi zawierające informacje o wymiarach konturów są tworzone podobnie, elementami ciągów są pary liczb określające wartości współrzędnych x oraz y końca każdego z odcinków jednostkowych opisujących kształt konturu.

Przykłady tworzenia takich ciągów, dla elementów pokazanych na rys. 4, ilustruje rys. 5.



Rys. 4. Przykładowy opis dwóch różnych konturów



Rys. 5. Przykład tworzenia ciągów zawierających informacje o wymiarach konturów (por. rys. 4)

Każdy ze stanów maszyny o skończonej liczbie stanów będzie opisany też dwoma typami ciągów, tworzonych zgodnie z powyższymi regułami. Reguły budowy ciągów T i TW są identyczne.

4.3. Opis czynności wykonywanych przez obrabiarkę

W omawianym przykładzie przyjęto model prostej obrabiarki, która może wykonywać 8 różnych czynności (instrukcji). Koszt zastosowania każdej z instrukcji jest przedstawiony w tablicy 1.

Tablica 1

Nr instrukcji μ	1	2	3	4	5	6	7	8
Koszt zastosowania	0,30	0,20	0,35	0,25	0,10	0,40	0,15	0,45

Każda z 8 instrukcji przykładowej obrabiarki powoduje zmianę kształtu konturu w taki sposób, że pole konturu ulega zmniejszeniu o pole kwadratu o boku równym odcinkowi jednostkowemu r. Jest to równoznaczne z zeszlifowaniem fragmentu obrabianego przedmiotu.

Opis poszczególnych instrukcji zawiera informacje, które z fragmentów danego konturu zostaną zmienione i w jaki sposób należy zmienić te części XO_i i XWO_i, które opisują dany fragment. Wykonanie danej instrukcji μ polega więc na analizowaniu ciągu XO - idąc od lewej ku prawej (zgodnie z osią x - por. rys. 4), a po wykryciu, że dany fragment ciągu XO_i można zmienić, zostaje on zmieniony, po czym analizowanie przebiega dalej.

Jeśli żadnego fragmentu ciągu XO_i nie można było zmienić, stosując daną instrukcję μ, zachodzi:

$$\Psi(X_i, \mu) = X_i$$

Oznacza to, że danej instrukcji nie można zastosować. Podobnie analizuje się ciąg XWO_i.

Jeśli instrukcję μ można było zastosować, ciągi XO_i oraz XWO_i zostaną przekształcone, dając w wyniku opis nowego stanu X_j maszyny o skończonej liczbie stanów:

$$\Psi(X_i, \mu) = X_j \neq X_i$$

W tablicy 2 zamieszczono wspomniane powyżej reguły stosowania instrukcji μ.

Tablica 2

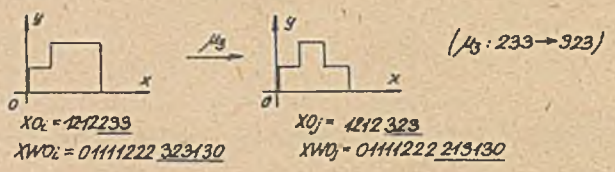
Nr instrukcji μ	Opis fragmentu ciągu XO	
	przed wykonaniem μ	po wykonaniu μ
1	112	121
2	121	211
3	233	323
4	323	332
5	122	212
6	212	221
7	223	232
8	232	322

Graficzne odwzorowanie tablicy 2 prezentuje rys. 6.

Zmiana odpowiedniego fragmentu ciągu XO_i narzuca jednocześnie sposób zmiany fragmentu ciągu XWO_i, na przykład: niech będzie dany następujący kontur (rys. 7). Po zastosowaniu instrukcji μ_3, której użycie było możliwe (wystąpiła bowiem kombinacja 233), otrzymano ciągi XO_j i XWO_j. Fragmenty, które uległy zmianie, zostały podkreślone. Łatwo zauważyć, że zmiana ciągu XWO_i polegała na dodaniu do odpowiedniego fragmentu tego ciągu,

pewnego ciągu c, charakterystycznego dla zastosowanej instrukcji. W tym przypadku był to ciąg $\{-1,-1,0,0,0,0\}$. Ciągi takie, charakterystyczne dla każdej instrukcji, zawiera tablica 3.

Rys. 6. Interpretacja graficzna instrukcji przykładowej obrabiarki



Rys. 7. Przykład wykonywania wybranej instrukcji obrabiarki

Tablica 3

Nr instrukcji μ	Postać ciągu C
1	0, 0, 1,-1,0,0
2	1,-1, 0, 0,0,0
3	-1,-1, 0, 0,0,0
4	0, 0,-1,-1,0,0
5	1,-1, 0, 0,0,0
6	0, 0, 1,-1,0,0
7	0, 0,-1,-1,0,0
8	-1,-1, 0, 0,0,0

4.4. Opis programu na maszynę cyfrową

Ogólny schemat blokowy programu na maszynę cyfrową serii ODRA 1300 pokrywa się ze schematem blokowym algorytmu przedstawionego na rys. 2.

Program został napisany w języku ALGOL 1900, posiada szereg dodatkowych sygnalizacji, które zostaną omówione w skrótowy sposób.

Działanie prezentowanego programu ATOS można opisać następująco:

Po wprowadzeniu danych liczbowych określających maksymalne wartości współrzędnych x oraz y, występujących w opisie ciągów XWO i TW, obliczona zostaje maksymalna możliwa ilość elementów w ciągu XO i T.

Podane w danych ciągi XO, T, XWO i TW są następnie badane - sprawdza się, czy zostały utworzone zgodnie z podanymi poprzednio regułami.

Następnie wczytywane są dane opisujące instrukcje μ - czyli tablice 1, 2, 3.

Utworzony w ten sposób opis zadania jest wydrukowany na drukarce wierszowej, po czym program zaczyna poszukiwać rozwiązania.

Informacje o dokonanym wyborze instrukcji w każdym stanie maszyny o skończonej liczbie stanów oraz przyczyny ewentualnych powrotów do poprzednich stanów są drukowane - pozwala to na zorientowanie się, w jaki sposób przebiegały próby tworzenia ciągu $\bar{\mu}$.

Jeśli rozwiązanie istnieje, zostaje wydrukowany ciąg $\bar{\mu}$ oraz wartość całkowitego kosztu przejścia ze stanu X_0 do stanu T - czyli koszt wykonania żądanego elementu.

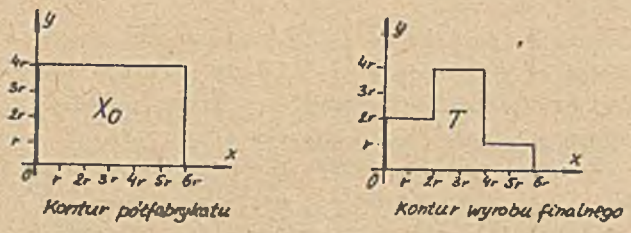
Jeśli natomiast zadanie jest nierozwiązalne, program drukuje stosowną informację.

W obu tych przypadkach oznacza to koniec przebiegu programu. Program ATOS składa się z szeregu procedur, które realizują fragmenty algorytmu, przedstawionego na rys. 2. I tak na przykład badanie, czy stan X_i maszyny o skończonej liczbie stanów jest taki, że wyklucza dojście do stanu T przeprowadza się, wykorzystując tę własność, że kontur opisany jako stan T musi dać się wpisać w kontur opisany jako stan X_i, aby dojście do stanu T było możliwe.

Ilustruje to dokładniej rysunek 8.

Rys. 8. Badanie możliwości osiągnięcia końcowego stanu maszyny M

Rozwiązanie przykładu, w którym kontur półfabrykatu oraz wyrobu finalnego pokazano na rysunku 9 - zamieszczono w tablicy 4.



Rys. 9. Opis stanu X_0 oraz stanu T w przykładzie rozwiązywanym przy pomocy programu ATOS

Tablica 4

Nr stanu	Nr instrukcji μ	Koszt zastosowania
1	5	0,10
2	5	0,10
3	7	0,15
4	7	0,15
5	1	0,30
6	3	0,35
7	3	0,35
8	3	0,35
9	4	0,25
10	6	0,40
11	stan końcowy	kontur T

Koszt całkowity: 2,50

Dodatkowo należy zaznaczyć, że program ATOS został napisany w ten sposób, że przy jego pomocy można rozwiązać zagadnienie znajdowania ciągu $\bar{\mu}$ dla dowolnie złożonego zadania postawionego zgodnie z zasadami podanymi w przykładzie.

Ograniczenie stanowi tu jedynie wielkość pamięci operacyjnej użytego do obliczeń komputera i wydłużający się czas obliczeń w przypadku długich ciągów XO i T.

(Dla porównania - czas obliczeń dla przykładu opisanego w pracy,w przypadku użycia ODRY 1325 wynosi ok. 1 minuty).

Kładąc w miejsce procedur użytych w krokach E1, E3, E5 i E6 algorytmu (rys. 2) inne procedury, można bez trudu adaptować program ATOS do rozwiązywania innych zadań.

5. Uwagi i wnioski końcowe

Przedstawiony w pracy algorytm pozwala na znalezienie rozwiązania postawionego zadania bez konieczności znajomości funkcji możliwości f. Użycie tego algorytmu może być zatem w pewnych przypadkach łatwiejsze niż użycie algorytmu podanego w pracy [1].

Rozwiązanie postawionego zadania zostaje znalezione zawsze - o ile istnieje, przy czym charakteryzuje się spełnieniem prostego kryterium optymalizacyjnego: przy przechodzeniu z każdego poprzedniego do następnego stanu maszyny M wybierana jest zawsze najtańsza instrukcja.

Zastosowanie takiego kryterium wydaje się być korzystne przy uwzględnieniu faktu, że ilość stanów pośrednich pomiędzy stanem XO i T jest nieznana i zależy od wyboru instrukcji μ w danym stanie, zaś ilość wszyst-

kich kombinacji, które należałoby sprawdzić może być bardzo duża. Ponadto nieznajomość funkcji możliwości nie pozwala na wyeliminowanie części kombinacji instrukcji. Program ATOS można w prosty sposób adaptować do opisu innego procesu, bądź też wykorzystywać jako makroinstrukcję rozmytą w złożonym programie rozmytym.

LITERATURA

[1] Jakubowski R., Kasprzak A.: Application of fuzzy programs to the desing of machining technology. Bull. Acad. Polon. Sci. Sèr.Techn.21/73.

[2] Jakubowski R., Kasprzak A.: Description and recognition of machine elements shapes. Bull. Acad. Polon. Sci. Sèr. Techn. 21/73.

[3] Chang S.H.: On the execution of fuzzy programs using finite-state machines. IEEE Trans. on Computers C-21 March 1972.

ПРИМЕНЕНИЕ МОДЕЛИ МАШИНЫ С КОНЕЧНЫМ ЧИСЛОМ ПОЛОЖЕНИЙ ДЛЯ ПРОЕКТИРОВАНИЯ АЛГОРИТМА УПРАВЛЕНИЯ ТЕХНИЧЕСКИМ ПРОЦЕССОМ

Р е з ю м е

В работе представлены рассуждения, касающиеся методов управления машиной с конечным числом положений (машиной М), таким образом, чтоб имея данные, касающиеся начального состояния, добиться конечного состояния, используя некоторый критерий оптимализации.

Представлены примеры для цифровой вычислительной машины, которая находит тягу информаций машины М, обеспечивающую доступ к заданному конечному положению при незнании общего числа состояний и возможности перехода с предыдущего положения в следующее.

Это даёт возможность формулировать задание в виде расплывчатой программы.

APPLICATION OF FINITE-STATE MACHINE MODEL TO THE DESIGN OF ALGORITHM FOR THE TECHNOLOGICAL PROCESS STEERING

S u m m a r y

In the paper there have been considered the methods of steering the finite-state machine (machine M) in such a way that having the data of the initial state we can obtain the final state, taking into consideration the simple optimalization criterion.

There has been presented an example of digital computer programme which finds the sequence of machine M instructions ensuring the access to a

given final state when the general number of states and possibilities to pass from the previous state to the next one is unknown. It allows the description of the problem in a fuzzy program form.