ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1997
Seria INFORMATYKA z 33 Nr kol. 1377

Adam MRÓZEK
Instytut Informatyki Teoretycznej i Stosowanej PAN
Marek SIKORA
Politechnika Śląska, Instytut Informatyki

SYNTEZA AUTOMATU MODELUJĄCEGO ZACHOWANIE SIĘ CZŁOWIEKA W PROCESIE PODEJMOWANIA DECYZJI

Streszczenie. Niniejszy artykuł zawiera przedstawienie problemu akwizycji, reprezentacji i utylizacji za pomocą komputera, wiedzy o „ludzkim" sposobie rozwiązywania określonych zadań. Podano algorytm konstrukcji automatu modelującego zachowanie się człowieka na podstawie obserwacji jego zachowań, zapisanych w postaci ustalonego protokołu, a także przykłady jego wykorzystania.

SYNTHESIS OF THE AUTOMATON MODELLING HUMAN`S BEHAVIOUR IN MAKING DECISIONS PROCESS

Summary. This article contains description of acquisition, representation and utilization with the aid of computer knowledge about „human" way of solving the definite tasks problem.There is also included algorithm of construction the automaton modelling human behaviour based on observing human`s behaviours, noting in form of established protocol and the examples of it`s application.

1. Wstęp

W informatyce kluczową rolę odgrywa pojęcie algorytmu. Mając algorytm możemy go zapisać w takim, czy innym języku programowania, a następnie zrealizować, za pomocą komputera, otrzymując rozwiązanie interesującego nas zadania, z określonej klasy zadań. Stąd, w dużym uproszczeniu można stwierdzić, że posiadanie odpowiednich algorytmów decyduje o skali zastosowań, środków i narzędzi informatycznych.

Tradycyjnie przez algorytm rozumie się uporządkowany, skończony ciąg operacji, których zastosowanie (wykonanie) do danych wejściowych prowadzi do otrzymania odpowiednich danych wyjściowych [1]. Tak określone algorytmy cechuje: masowość, determinizm i rezultatywność. Zarówno dane wejściowe, przetwarzane za pomocą algorytmu, jak i dane wyjściowe w sposób pełny i jednoznaczny charakteryzowały konkretne zadanie i odpowiadające mu rozwiązanie z danej klasy zadań.

Tak rozumiane pojęcie algorytmu oraz związanego z nim procesu obliczeniowego, realizowanego za pomocą komputera, nosi nazwę *hard computing* [2].

W miarę rozwoju i upowszechniania się metod, środków i narzędzi informatycznych okazało się, że istotnym ograniczeniem ich szerokich zastosowań jest konieczność pozyskiwania nowych algorytmów, pozwalających na opracowanie i implementację odpowiednich programów użytkowych.

Jedną z istotnych możliwości rozwiązania tego problemu jest zwrócenie uwagi na człowieka rozwiązującego złożone i różnorodne zadania. Z możliwością tą wiąże się konieczność odpowiedzi na następujące pytania: jak przejmować wiedzę o „ludzkim" sposobie rozwiązywania zadań , jak ją reprezentować za pomocą algorytmów, które następnie mogłyby być zapisane w języku programowania.

Okazuje się, że odpowiedź na te pytania jest trudna. Wiedza, którą posiada i wykorzystuje człowiek, jest w znacznej mierze intuicyjna. Charakteryzuje ją nieostrość, niepełność, a czasami także niepewność. Również swoje zachowanie się, np. przy podejmowaniu określonych decyzji, człowiek nie zawsze może w sposób jednoznaczny uzasadnić. Prowadzi to do konkluzji, że przy próbach algorytmicznego ujęcia zachowania się człowieka w procesie podejmowania różnorodnych decyzji, tradycyjne rozumienie algorytmu może okazać się niewystarczające. Świadomość tego faktu stanęła u podstaw określenia nowego kierunku badań w informatyce nazwanego *soft computing* [2], który integruje liczne i różnorodne wcześniejsze prace, związane z próbami przejmowania i praktycznego wykorzystania wiedzy człowieka w tzw. systemach eksperckich, a także wytycza nowe kierunki badań.

Celem artykułu jest przedstawienie pewnej propozycji podejścia do rozwiązania problemu akwizycji, reprezentacji i utylizacji, za pomocą komputera, wiedzy o „ludzkim" sposobie rozwiązywania określonych zadań.

2. Przedstawienie zachowania się człowieka

Wspomniane powyżej rozumienie algorytmu może być wyrażone za pomocą paradygmatu przedstawionego w [3] w postaci:

< algorytm >=< logika >+< sterowanie > (1)

W zapisie tym składnik <logika> reprezentuje dostępną wiedzę o sposobie rozwiązywania zadania, zaś składnik <sterowanie> reprezentuje znane strategie wykorzystania tej wiedzy w procesie rozwiązywania zadania za pomocą algorytmu.

Wykorzystując taką samą reprezentację algorytmów opisujących „ludzki" sposób rozwiązywania zadań, należy odpowiedzieć na pytanie, co będą reprezentowały i jaką będą miały postać składniki <logika> i <sterowanie>.

2.1. Regułowa reprezentacja wiedzy

Jak to wykazano między innymi w [4], naturalnym sposobem reprezentacji wiedzy człowieka (np. przy sterowaniu złożonymi obiektami technologicznymi, diagnostyce medycznej), jest wykorzystanie tzw. produkcji warunkowej (reguły decyzyjnej) postaci:

IF { zbiór warunków } THEN { zbiór decyzji (czynności)}

Zbiór takich produkcji warunkowych, otrzymanych np. od człowieka będącego ekspertem w danej dziedzinie lub operatorem procesu technologicznego, wykorzystywanych w trakcie rozwiązywania zadania, z danej klasy zadań, można zebrać tworząc np. regułową bazę wiedzy. Stąd, składnik < logika >, występujący w (1), będzie dalej utożsamiany z pojęciem < regułowa baza wiedzy > co można symbolicznie zapisać w postaci:

< logika >≡< regułowa baza wiedzy >

2.2. Mechanizm wyboru reguły decyzyjnej

Przyjmując, że cała dostępna wiedza o podejmowania decyzji przy rozwiązywaniu zadań z określonej klasy zadań, jaką posiada człowiek, zapisana jest w regulowej bazie wiedzy, duże znaczenie ma podanie efektywnego sposobu wyboru sekwencji odpowiednich reguł decyzyjnych, w przypadku rozwiązywania konkretnego zadania.

Naturalnym i jak dotąd, najczęściej stosowanym sposobem postępowania może być metoda przeszukiwania regułowej bazy wiedzy. W najprostszym ujęciu metoda przeszukiwania polega na sprawdzaniu warunków wyspecyfikowanych w każdej regule decyzyjnej i przejściu do realizacji czynności wyspecyfikowanych dla tej reguły decyzyjnej, dla której wszystkie warunki zostały spełnione.

Oczywiście, taki sposób postępowania nie jest jednoznaczny. Co na przykład zrobić w sytuacji, gdy wszystkie warunki są spełnione dla kilku reguł decyzyjnych lub gdy nie są spełnione dla żadnej reguły decyzyjnej? Innym problemem jest efektywność takiego przeszukiwania w przypadku, gdy regułowa baza wiedzy zawierać będzie dużą liczbę reguł decyzyjnych [6].

Nie wchodząc w szczegóły odpowiedzi na tego rodzaju pytania, przyjmujemy, że składnik < sterowanie >, występujący w (1) będzie dalej utożsamiany z pojęciem < mechanizm wyboru reguły decyzyjnej >, co można symbolicznie zapisać:

< sterowanie >≡< mechanizm wyboru reguły decyzyjnej >

2.3. Reprezentacja „ludzkich" algorytmów

Wykorzystując reprezentację algorytmu w postaci paradygmatu (1) oraz ustalone powyżej znaczenie jego poszczególnych składników można przyjąć następujący sposób zapisu „ludzkiego" algorytmu:

< algorytm >=< regułowa baza wiedzy >+< mechanizm wyboru reguły decyzyjnej >

Reprezentacja zachowania się człowieka przy rozwiązywaniu zadań wymaga zatem:

a) podania sposobu pozyskiwania reguł decyzyjnych i syntezy regułowej bazy wiedzy;
b) ustalenia mechanizmu wyboru odpowiednich reguł decyzyjnych.

2.3.1. Metody pozyskiwania reguł decyzyjnych i syntezy regułowej bazy wiedzy

Najbardziej znanymi sposobami pozyskiwania reguł są: wywiady z ekspertami, modele matematyczne oraz maszynowe uczenie się [5]. Z innych interesujących metodologii pozyskiwania wiedzy można wymienić akwizycję reguł decyzyjnych opartą na teorii zbiorów przybliżonych [4].

Ponieważ systemy wyposażone w możliwości uczenia się generują nową wiedzę, duży nacisk kładzie się na jakość tworzonej w ten sposób bazy wiedzy (podobnie jest w przypadku wiedzy uzyskanej od eksperta). Dlatego też istotne znaczenie ma badanie poprawności bazy wiedzy (walidacja). W przypadku regułowej bazy wiedzy przeprowadza się m.in. badanie spójności bazy wiedzy, eliminuje się niepotrzebne warunki i reguły pochłaniające.

Metodologia tworzenia i walidacji regułowej bazy wiedzy, oparta na formalizmie teorii zbiorów przybliżonych, opisana została w [6].

Przyjmujemy, że baza wiedzy reprezentowana jest w postaci tablicy decyzyjnej [7] lub tablicy obserwacji [4] w następującej postaci:

Tabela 1

Tablica decyzyjna

Numer reguły	Atrybuty warunkowe					Atrybuty Decyzyjne				
	c_1	...	c_j	...	c_k	d_1	...	d_i	...	d_p
1	$v^1_{c_1}$	...	$v^1_{c_j}$	...	$v^1_{c_k}$	$v^1_{d_1}$	...	$v^1_{d_i}$	...	$v^1_{d_p}$
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.
n	$v^n_{c_1}$	...	$v^n_{c_j}$	...	$v^n_{c_k}$	$v^n_{d_1}$	...	$v^n_{d_i}$	...	$v^n_{d_p}$

W tablicy takiej możemy zapisać wiedzę eksperta lub po odpowiedniej modyfikacji, obserwacje działań operatora procesu technologicznego. Pomijając problem dyskretyzacji wartości, jakie mogą przyjmować poszczególne atrybuty, tablicę taką możemy minimalizować i dzielić na część spójną i niespójną wykorzystując wspomnianą już teorię zbiorów przybliżonych [8]

2.3.2. Mechanizm wyboru reguły decyzyjnej

Jak już wspomniano wcześniej, jedną z najbardziej popularnych metod wyboru reguły decyzyjnej jest metoda przeszukiwania regułowej bazy wiedzy. Niestety, do tej pory nie powstała autonomiczna i w pełni zadowalająca strategia sterująca mechanizmem wyboru reguły decyzyjnej.

Mechanizm wyboru reguły decyzyjnej może być również określany przez człowieka, np. przez programistę lub eksperta dziedziny, który korzystając z dostępnych w bazie wiedzy faktów i operatorów (w naszym przypadku reguł decyzyjnych), rozwiązuje postawiony problem, czyli umiejętnie wykorzystuje dostępną wiedzę .

Nie zawsze mechanizm wyboru reguły decyzyjnej może być w pełni wyspecyfikowany. Dlatego prowadzi się badania dotyczące jego syntezy na podstawie obserwacji lub przykładów zachowania się człowieka (lub ogólniej systemu), którego takie zachowanie się chcemy „przejąć".

Poniżej przedstawiono metodę syntezy mechanizmu wyboru reguły decyzyjnej opartej na pojęciu deterministycznego systemu produkcyjnego (DPS) oraz realizującego go automatu [8].

2.4. Sformułowanie problemu

Niech $P=\{p_1, p_2, .., p_n\}$ będzie zbiorem reguł decyzyjnych występujących w regułowej bazie wiedzy.

Ustalając protokół zachowania się człowieka jako ciąg kolejno wykorzystywanych przez niego reguł decyzyjnych:

$t=(p_0, p_1, .., p_s, p_e)$, gdzie $p_j \in P$; $j=1,2,..,s$ $s \leq n$; p_0, p_e reguły pomocnicze (separatory) ograniczające ciągi reguł decyzyjnych; (2)

możemy sformułować interesujący nas problem następująco:

mając dany zbiór $S=\{t_1, t_2, .., t_m\}$, którego elementami są ciągi postaci (2) będące przykładami zachowania się człowieka, skonstruować automat rozpoznający wszystkie elementy zbioru S (modelujący wszystkie przykładowe zachowania).

3. Metoda syntezy automatu modelującego zachowanie człowieka

Opierając się na przyjętych założeniach co do sposobu reprezentacji wiedzy (regułowa baza wiedzy zapisana w postaci tablicy decyzyjnej), protokołu zachowania się człowieka (sekwencja kolejnych zastosowań reguł uzyskanych w czasie obserwacji działań człowieka) oraz na pracy [8] przedstawimy metodę syntezy automatu modelującego zachowanie się człowieka opisane przykładami jego działań.

Definicja.1 Deterministyczny System Produkcyjny DPS to uporządkowana czwórka (P, δ, ϕ, p_s), gdzie $P=\{p_1, .., p_n\}$ - zbiór reguł decyzyjnych; p_s - reguła startowa; δ i ϕ - tzw. pola sterowania.

Trójkę $(p_i, \delta(p_i), \phi(p_i))$ interpretuje się następująco:

IF warunki reguły decyzyjnej p_i są spełnione THEN zastosuj regułę decyzyjną p_i; sprawdź, czy spełnione są warunki reguły decyzyjnej $\delta(p_i)$; (3)

ELSE sprawdź, czy spełnione są warunki reguły decyzyjnej $\phi(p_i)$.

Ponadto dana jest reguła stopu λ, której zastosowanie zatrzymuje pracę DPS.

Automat realizujący konkretny DPS postaci (P, δ, ϕ, p_s), może być przedstawiony jak poniżej:

	δ	ϕ
p_1	$\delta(p_1)$	$\phi(p_1)$
p_2	$\delta(p_2)$	$\phi(p_2)$
:	:	:
p_n	$\delta(p_n)$	$\phi(p_n)$

Definicja.2 Zbiór $S(p)=\{p_i$: reguła decyzyjna p_i została użyta bezpośrednio po regule decyzyjnej $p\}$ nazywamy następnikiem reguły decyzyjnej $p\in P$.

Przez zbiór $F^t(p_i, p_j)$ i=0,1,..,n j=1,2,..,n,e oznaczymy najmniejszy zbiór reguł decyzyjnych, które nie mogą być zastosowane po użyciu reguły decyzyjnej p_i przed wcześniejszym wykorzystaniem reguły decyzyjnej p_j w przykładzie t.

Algorytm wyznaczania zbioru $F^t(p_i, p_j)$ jest następujący:

1 Jeżeli $p_i p_j$ nie występują jedna po drugiej w przykładzie t, **to** $F^t(p_i, p_j)=P$;

2 Jeżeli i=0, **to** $F^t(p_i, p_j)=\{ p_k$: p_k nie występuje przed p_j w przykładzie t $\}$;

3 Jeżeli j=e, **to** $F^t(p_i, p_j)=\{ p_k$: p_k nie występuje po p_i w przykładzie t $\}$;

4 Jeżeli i,j=1,..,n, **to** $F^t(p_i,p_j)=\{ p_k$: p_k nie występuje bezpośrednio po regule decyzyjnej p_i w przykładzie t $\}\cap\{p_k$: p_k występuje bezpośrednio po regule decyzyjnej p_j w przykładzie t $\}$.

Definicja.3 $F(p_i, p_j)=\bigcap_{t_k\in S} F^{t_k}(p_i, p_j)$ dla i=0,1,..,n j=1,..,n,e.

Definicja.4 Dla każdego zbioru S(p), zbiory reguł $E(p)=\{p_1, .. ,p_k\}$ ustawionych w porządku $V(p)=(p_1, .. , p_k)$, spełniającym następujące warunki:

a) $p_i\neq p_j \quad \forall i\neq j$ i,j=1,2,..,k;

b) $p_k\in S(p)$;

nazywamy S(p)-zakończonymi.

Niech E(P) będzie zbiorem zawierającym zbiory S(p)-zakończone dla wszystkich S(p) wyliczonych z przykładów.

Definicja.4 Zbiór E(P) nazywamy S-niesprzecznym wtedy i tylko wtedy, gdy dla każdych $E(r), E(p) \in E(P)$ istnieją porządki $V(r)=(r_1 , .. ,r_k)$ i $V(p)=(p_1 , .. ,p_m)$ odpowiednio takie, że jeżeli $r_i=p_j$ dla pewnych $1\leq i\leq k$, $1\leq j\leq m$, to wtedy

albo $(r_i ,r_{i+1} , .. ,r_k)$ jest podciągiem $(q_j ,q_{j+1} , .. ,q_m)$;

albo $(q_j ,q_{j+1} , .. ,q_m)$ jest podciągiem $(r_i ,r_{i+1} , .. ,r_k)$.

Zbiór reguł $P\cup\{p_e\}$ można podzielić relacją zwrotną i symetryczną ρ określoną na iloczynie kartezjańskim tego zbioru następująco:

$(r_i,r_j)\in\rho \Leftrightarrow \exists$ r,p $\in$P takie, że $r_i\in S(r)$, $r_j\in S(p)$ oraz $S(r)\cap S(p)\neq\emptyset$.

Przechodnie domknięcie relacji ρ indukuje funkcję $h_\rho: S\rightarrow\Pi$ przekształcającą następnik reguły $S=\{ S(p) : p\in P \}$ w określony blok podziału $\Pi=P\cup\{p_e\}/\rho$ zbioru P. Można również wskazać te S(p), które dla określonego bloku podziału $\pi\in\Pi$ spełnieją zależność $S(p)\in h_\rho^{-1}(\pi)$.

Definicja.5 Dla ustalonego bloku $\pi\in\Pi$ $F_\pi(p_i, p_j)= F(p_i, p_j)\cap\pi$ dla i=0,1,..,n j=1,..,n,e.

Dla każdego bloku π podziału Π szukamy S-niesprzecznego zbioru dla $S(p)\in h_\rho^{-1}(\pi)$, aby wyznaczyć zawartość pól sterowania dla reguł pojawiających się w bloku π.

Wyznaczamy dla wszystkich reguł $p \in \pi$ zbiory $\overline{\Delta}_\pi(p)$ oraz $\overline{\Phi}_\pi(p)$, które będą estymatorami dla $\delta(p)$, $\phi(p)$ w tym bloku podziału:

$\overline{\Delta}_\pi(p)=\{p_m\} \cup F_\pi(p, p_m)$, jeżeli $|F_\pi(p, p_m)| \leq |F_\pi(p, p_j)|$ dla każdej $p_j \in \pi$, $p_m \in P$.

Zbiór $\overline{\Phi}_\pi(p)$ definiujemy następująco, jeżeli $S(p_i) \in h^{-1}(\pi)$ i $p \in S(p_i)$ wtedy

$\phi p_i(p)=\{ p_k : |F_\pi(p_i, p_k)| \leq |F_\pi(p_i, p_j)|$ dla każdej $p_j \in \pi$ takiej, że $p \in F_\pi(p_i, p_j) \}$

Ostatecznie $\overline{\Phi}_\pi(p)=\bigcup \phi p_i(p)$ (sumujemy po takich p_i, że $S(p_i) \in h^{-1}(\pi)$ i $r \in S(p_i)$).

Stwierdzenie 1. Prawdziwe są następujące zależności:

$$\delta(p) \in \overline{\Delta}_\pi(p); \qquad \phi(p) \in \overline{\Phi}_\pi(p).$$

Wyznaczamy elementy zbiorów $LV_\pi(p_i)$, które są zbiorami porządków S(p)-zakończonych w $\pi=\{r_1, r_2, .., r_k\}$, kandydującymi do wybrania z nich zbioru S-niesprzecznego. Zbiory te wyliczamy następująco:

definiujemy dwie macierze

Σ i $\Phi \in M^{k*k}$ gdzie: $\Sigma=[\delta_{ij}]$ gdzie $\delta_{ij}=\{p_j\}$ jeżeli $p_j \in \overline{\Delta}_\pi(p)$; w przeciwnym wypadku $\delta_{ij}=\emptyset$;

$\Phi=[\phi_{ij}]$ gdzie $\phi_{ij}=\{p_j\}$ jeżeli $p_j \in \overline{\Phi}_\pi(p)$; w przeciwnym wypadku $\phi_{ij}=\emptyset$

a następnie kolejne macierze $\Psi^{(0)}=\Sigma$; $\Psi^{(i)}=\Psi^{(i-1)}\Phi$ dla i=1, .. ,k-1 oraz $\Psi_i=\bigcup_{j=1}^{k} \Psi_{ij}{}^{(k-1)}$,

gdzie $\Psi_{ij}{}^{(k-1)}$ jest ij-tym elementem $\Psi^{(k-1)}$ (produkt macierzy zgodny z def. podaną w [8]).

Stwierdzenie.2 Dla każdego i=1, .. ,k, $\Psi_i \supseteq LV_\pi(p_i)$.

Opierając się na wprowadzonych definicjach i stwierdzeniach oraz na pracy [8] można podać następujący algorytm syntezy automatu modelującego zachowanie się człowieka:

Algorytm

INPUT : regułowa baza wiedzy $P=\{p_1, .. , p_n\}$, zbiór przykładów $S=\{t_1, .., t_m\}$;

OUTPUT: automat (P, δ, ϕ, p_s) rozpoznający wszystkie elementy S.

1. Wyznacz wszystkie zbiory $S(p_i)$, i=0..n.

2. Wyznacz podział $P \cup \{p_e\}/\rho$.

3. Określ, przeciwdziedzinę funkcji $h_\rho: S \rightarrow P \cup \{p_e\}/\rho$ (tzn. $\forall p_i \in P$ określ taki blok π z podziału $P \cup \{p_e\}/\rho$, że $S(p_i) \in h_\rho^{-1}(\pi)$)

4. Wyznacz zbiory $F'(p_i, p_j)$, i=0..n, j=1..n,e dla każdego z przykładów oraz zbiory $F(p_i, p_j)$

5. Ustal blok podziału π

5.1. Jeżeli blok jest jednoelementowy (czyli $\pi=\{p\}$), **to** $LV_{[p]}=\{ (p) \}$ **idź do 5.4.**

5.2. Jeżeli $\pi=\{p_1,..,p_m\}$ $m \leq n$ **to**

5.2.1. dla wszystkich i=1..m wyznacz $\bar{\Delta}_\pi(p_i)$ i $\bar{\Phi}_\pi(p_i)$;

(dla p_e $\bar{\Delta}_\pi(p_e)=\emptyset$ i $\bar{\Phi}_\pi(p_e)=\emptyset$)

5.2.2. wypełnij macierze Σ i Φ;

5.2.3. wylicz macierze Ψ^0 .. Ψ^{m-1};

5.2.4. na podstawie macierzy Ψ^{m-1} wyznacz zbiory $LV_\pi(p_i)$ dla i=1..m;

5.2.4.1. i-ty wiersz macierzy Ψ^{m-1} odpowiada zbiorowi $LV_\pi(p_i)$, po uprzednim usunięciu z niego tych elementów, które nie są $S(p_i)$-zakończone (elementami tego zbioru są ciągi reguł). Jeżeli $\pi \cap S(p_i)=\emptyset$ to z i-tego wiersza macierzy Ψ^{m-1} nie usuwamy żadnych elementów.

5.3. Dla każdego bloku π ustaw zbiory LV_π w porządku $LV_\pi(p_{i_1}), .., LV_\pi(p_{i_m})$ takim, że $|LV_\pi(p_{i_j})| \leq |LV_\pi(p_{i_{j+1}})|$.

5.4. Wybierz S-niesprzeczne porządki $V(p_i)$ dla wszystkich $S(p_i) \in h_\rho^{-1}(\pi)$ stosując następującą zasadę:

1) porządki wybieramy począwszy od $LV_\pi(p_{i_1})$;

2) wybrany porządek $V(p_i)=(p_{i_1},..,p_{i_j})$ musi spełniać warunki

$S(p_i) \subseteq \{p_{i_1},..,p_{i_j}\}$ oraz $p_{i_j} \in S(p_i)$;

3) w skład wszystkich S-niesprzecznych porządków $V(p_i)$ dla wszystkich $S(p_i) \in h_\rho^{-1}(\pi)$ muszą wchodzić wszystkie reguły z bloku π.

6. Jeżeli wykonałeś 5 dla wszystkich bloków, **to idź do 7, w przeciwnym razie idź do 5.**

7. Mając wybrane S-niesprzeczne porządki $V(p_i)$ dla wszystkich i=0,..,n wyznacz wartości funkcji δ i ϕ oraz regułę startową:

7.1. regułą startową jest pierwsza reguła w porządku $V(p_0)$.

7.2. dla każdego $V(p)=(p_1, .. ,p_k)$ $p \neq p_0$

$$\delta(p)=\begin{cases} p_1 & \text{jeżeli } p_1 \neq p_e; \\ \lambda & \text{jeżeli } p_1 = p_e. \end{cases}$$

7.3. dla każdej $p \in P$

$$\phi(p)=\begin{cases} r & \text{jeżeli } pr \text{ jest podciągiem pewnego } V(p_i) \text{ oraz } r \neq p_e; \\ \lambda & \text{jeżeli } pp_e \text{ jest podciągiem pewnego } V(p_i) \text{ lub } p \text{ pojawia się tylko jako ostatni element w pewnym } V(p_i). \end{cases}$$

8. Koniec

4. Przykłady ilustrujące działanie wprowadzonego algorytmu

Podamy trzy przykłady ilustrujące działanie algorytmu. Przykłady mają na celu prezentację przydatności proponowanego podejścia do syntezy automatu modelującego zachowanie się eksperta w różnorodnych sytuacjach decyzyjnych.

4.1. Dodawanie liczb w słupku

Dodając liczby w słupku wykorzystujemy, w zasadzie, dwie reguły decyzyjne, których części warunkowe mają następującą postać (w celu uproszczenia nie będziemy rozpatrywać części decyzyjnej reguł, ponieważ dla naszych celów jest to niepotrzebne):

p_1 : suma liczb w kolumnie $\geq 10 \rightarrow$ działanie1;

p_2 : suma liczb w kolumnie $< 10 \rightarrow$ działanie2.

Obserwujemy dwa przykłady dodawania liczb

1046	602
+ 8972	+535
10018	1137

Odpowiadają im dwa ciągi stosowanych reguł decyzyjnych (p_0, p_2, p_1, p_1, p_1, p_e), (p_0 ,p_2 ,p_2 ,p_1 ,p_e).

$P=\{p_1, p_2\}$, p_0, p_e

1. $S(p_0)=\{p_2\}$ $S(p_1)=\{p_1, p_e\}$ $S(p_2)=\{p_1, p_2\}$

2.$P\cup\{p_e\}/\rho=[p_1, p_2, p_e]$

3. $S(p_0),S(p_1),S(p_2)\in h_\rho^{-1}([p_1,p_2,p_e])$

4.Na podstawie obu przykładów wyznaczamy zbiory $F(p_i,p_j)$ i=0,1,2 j=1,2,e;

F	p_1	p_2	p_3
p_0	P	p_1	P
p_1	∅	P	p_2
p_2	∅	∅	P

5. Relacja ρ wyznacza tylko jeden blok podziału zbioru $P\cup\{p_e\}$, jest nim $[p_1 ,p_2 ,p_e]$

5.2.1.

$\bar{\Delta}_{[p_1,p_2,p_e]}(p_1) = \{ p_1 \}$; $\bar{\Phi}_{[p_1,p_2,p_e]}(p_1) = \{ p_1, p_2, p_e \}$;

$\bar{\Delta}_{[p_1,p_2,p_e]}(p_2) = \{ p_1, p_2 \}$; $\bar{\Phi}_{[p_1,p_2,p_e]}(p_2) = \{ p_1, p_2, p_e \}$;

$\bar{\Delta}_{[p_1,p_2,p_e]}(p_e) = \varnothing$; $\bar{\Phi}_{[p_1,p_2,p_e]}(p_e) = \varnothing$.

5.2.2. Otrzymujemy dwie macierze:

$$\Sigma = \begin{bmatrix} p_1 & \varnothing & \varnothing \\ p_1 & p_2 & \varnothing \\ \varnothing & \varnothing & \varnothing \end{bmatrix}; \qquad \Phi = \begin{bmatrix} p_1 & p_2 & p_e \\ p_1 & p_2 & p_e \\ \varnothing & \varnothing & \varnothing \end{bmatrix}.$$

5.2.3. i 5.2.4. Wyliczamy macierze Ψ^1, Ψ^2 w efekcie otrzymamy:

$LV_{[p_1,p_2,p_e]}(p_1) = \{(p_1,p_e),(p_1),(p_e),(p_2,p_e),(p_1,p_2,p_e)\}$;

$LV_{[p_1,p_2,p_e]}(p_2) = \{(p_1),(p_2),(p_2,p_1),(p_1,p_2)\}$;

$LV_{[p_1,p_2,p_e]}(p_e) = \{(p_2,p_1),(p_e),(p_1,p_e),(p_1,p_e),(p_2,p_e)\}$;

5.3. $|LV_{[p_1,p_2,p_e]}(p_2)| \le |LV_{[p_1,p_2,p_e]}(p_e)| = |LV_{[p_1,p_2,p_e]}(p_1)|$.

5.4. Wybieramy porządki $V(p_0)=(p_2)$, $V(p_1)=(p_1,p_e)$, $V(p_2)=(p_2,p_1)$.

7.Mając wybrane porządki wyznaczamy wartości funkcji δ i ϕ. Otrzymujemy wywnioskowany automat.

	δ	ϕ
p_1	p_1	λ
p_2 - startowa	p_2	p_1

4.2. Problem Emdena i Kowalskiego

Rozpatruje się zbiór następujących podstawień: (w:=1), (wynik:=w), ($u:=u^2$;v:=v/2), (v:=v-1;w:=u*w). Użycie każdego z tych podstawień uzależnione jest od pewnych warunków, co pozwala zapisać cztery reguły decyzyjne:

p_1: null $\rightarrow$ w:=1; p_2: (v mod 2)=0 $\rightarrow$ $u:=u^2$; v:=v/2; p_3: v <> 0 $\rightarrow$ v:=v-1 ; w:=w*u;

p_4: v=0 $\rightarrow$ wynik:=w.

Obserwujemy dwa przykłady wykonywania działań wykorzystujących powyższe reguły decyzyjne. Możemy działania te zapisać w tablicy:

Tabela 2

Dwa przykłady zachowań dla problemu Emdena i Kowalskiego

Zachowanie pierwsze					Zachowanie drugie				
Reguła	u	v	w	wynik	Reguła	u	v	w	wynik
	2	10				2	10		
p_1	2	10	1		p_1	2	10	1	
p_2	4	5	1		p_2	4	5	1	
p_3	4	4	4		p_3	4	4	4	
p_2	16	2	4		p_2	16	2	4	
p_2	256	1	4		p_3	16	1	64	
p_3	256	0	1024		p_3	16	0	1024	
p_4				1024	p_4				1024

Otrzymaliśmy zatem dwa ciągi użytych reguł decyzyjnych będące zapisami działań podanych w tablicy (p_0, p_1, p_2, p_3, p_2, p_2, p_3, p_4, p_e), (p_0, p_1, p_2, p_3, p_2, p_3, p_3, p_4, p_e).

$P=\{p_1,\ldots,p_4\}, p_0, p_e$.

1. $S(p_0)=\{p_1\}$; $S(p_1)=\{p_2\}$; $S(p_2)=\{p_2, p_3\}$; $S(p_3)=\{p_2, p_3, p_4\}$ $S(p_4)=\{p_e\}$.

2. $P/\rho=[p_1]\cup[p_2, p_3, p_4]\cup[p_e]$.

3. $S(p_0)\in h_\rho^{-1}([p_1])$; $S(p_1), S(p_2), S(p_3)\in h_\rho^{-1}([p_2, p_3, p_4])$; $S(p_4)\in h_\rho^{-1}([p_e])$.

4. Wyliczamy zbiory $F(p_i,p_j)$; i=0,1,2,3,4, j=1,2,3,4,e;

F	p_1	p_2	p_3	p_4	p_5
p_0		P	P	P	P
p_1	P	p_3	P	P	P
p_2	P	∅	p_4	P	P
p_3	P	∅	∅	∅	P
p_4	P	P	P	P	

Podobnie jak w przykładzie pierwszym, wyliczamy estymatory funkcji ϕ i δ oraz zbiory $LV(p_i)$ dla każdego bloku podziału Π.

5. Bloki jednoelementowe $[p_1]$, $[p_e]$:

5.1. $LV_{[p_1]}=\{(p_1)\}$, $LV_{[p_e]}=\{(p_e)\}$.

5.4. $V(p_0)=(p_1)$, $V(p_4)=(p_e)$.

5. Blok $[p_2, p_3, p_4]$:

5.2.1.

$\bar{\Delta}_{[p_2,p_3,p_4]}(p_2) = \{ p_2 \}$; $\bar{\Phi}_{[p_2,p_3,p_4]}(p_2) = \{ p_1, p_2, p_3, p_4 \}$;

$\bar{\Delta}_{[p_2,p_3,p_4]}(p_3) = \{ p_2, p_3, p_4 \}$; $\bar{\Phi}_{[p_2,p_3,p_4]}(p_3) = \{ p_1, p_2, p_3, p_4 \}$;

$\bar{\Delta}_{[p_2,p_3,p_4]}(p_4) = \{ p_1, p_2, p_3, p_4 \}$; $\bar{\Phi}_{[p_2,p_3,p_4]}(p_4) = \varnothing$.

5.2.2.

$$\Sigma = \begin{bmatrix} p_2 & \varnothing & \varnothing \\ p_2 & p_3 & p_4 \\ p_2 & p_3 & p_4 \end{bmatrix}; \qquad \Phi = \begin{bmatrix} p_2 & p_3 & p_4 \\ p_2 & p_3 & p_4 \\ \varnothing & \varnothing & \varnothing \end{bmatrix}.$$

5.2.3. i 5.2.4. Następnie wyliczamy macierze Ψ^1, Ψ^2. W efekcie otrzymamy:

$LV_{[p_2,p_3,p_4]}(p_2) = \{ (p_3,p_2),(p_2),(p_2,p_3) \}$; $LV_{[p_2,p_3,p_4]}(p_3) = \{ (p_2),(p_3),(p_4),(p_3,p_2),$ $(p_4,p_2),(p_2,p_4),(p_3,p_4),(p_2,p_3),(p_4,p_3),(p_3,p_2,p_4),(p_2,p_3,p_4) \}$;

$LV_{[p_2,p_3,p_4]}(p_4) = LV_{[p_2,p_3,p_4]}(p_3)$.

5.3. $|LV_{[p_2,p_3,p_4]}(p_2)| \le |LV_{[p_2,p_3,p_4]}(p_3)| = |LV_{[p_2,p_3,p_4]}(p_4)|$

5.4. Wybieramy S-niesprzeczne porządki dla zbiorów $S(p_i) \in h_p^{-1}([p_2, p_3, p_4])$,

$V(p_1)=(p_2)$, $V(p_2)=(p_2,p_3)$, $V(p_3)=(p_2,p_3,p_4)$.

7. Wyznaczamy wartości funkcji δ i ϕ. Otrzymujemy wywnioskowany automat.

	δ	ϕ
p_1 - startowa	p_2	λ
p_2	p_2	p_3
p_3	p_2	p_4
p_4	λ	λ

Ponieważ w przykładach, od pewnego momentu, wybierana była dowolna reguła decyzyjna, otrzymujemy automat, który modeluje jedno z zachowań, ale produkuje poprawny wynik. Można by dodatkowo wprowadzić np. preferencje co do kolejności stosowania reguł decyzyjnych (w przypadku gdy prowadziłyby one do tych samych wyników), wtedy wybieralibyśmy rozszerzenia z preferowanymi regułami decyzyjnymi. Prowadziłoby to do wyznaczania automatu modelującego zachowanie za pomocą preferowanych przez nas reguł decyzyjnych.

Ponieważ automat konstruuje się dla określonego celu, możemy próbować wybierać jednoznacznie rozszerzenia stosując np. kryterium jak najszybszego osiągnięcia tego celu;

dodając stany, w jakich znajduje się system, można także zażądać utrzymania systemu najdłużej w jakimś stanie lub najszybszego osiągnięcia danego stanu.

4.3. Modelowanie zachowania palacza pieca obrotowego do wypalania klinkieru

Model i opis pieca do wypalania klinkieru zaczerpnięto z pracy [4], tam też można znaleźć charakterystykę wszystkich zmiennych i atrybutów.

W polu naszego zainteresowania jest palacz (ekspert) pieca, a dokładniej, „przejęcie" jego zachowania w trakcie sterowania piecem obrotowym. Palacz określił stany charakterystyczne, w jakich może znajdować się piec obrotowy i odpowiednie wartości atrybutów warunkowych w każdym stanie charakterystycznym. Zachowanie palacza opisane było zgodnie z wcześniej wypełnioną tablicą obserwacji. Powstała w ten sposób tablica decyzyjna, która przedstawiała zachowanie palacza w odstępach sześciominutowych. Przeanalizowano dwa ciągi przykładów zachowania palacza. W celu uproszczenia przykładu rozpatrzono tylko siedem kolejnych zachowań palacza, przedstawionych w poniższej tabeli:

Tabela 3

Tablica obserwacji zachowań palacza

Pierwszy przykład zachowań						Drugi przykład zachowań					
stan	nr reguły	wartości atrybutów warunkowych		wartości atrybutów decyzyjnych		stan	nr reguły	wartości atrybutów warunkowych		wartości atrybutów decyzyjnych	
		c_1	c_5	d_1	d_2			c_1	c_5	d_1	d_2
1	p_1	2	1	2	4	1	p_1	2	1	2	4
2	p_2	2	2	2	3	2	p_3	3	2	2	3
3	p_3	3	2	2	3	3	p_6	3	4	2	3
3	p_4	2	1	2	3	3	p_4	2	1	2	3
1	p_1	2	1	2	4	1	p_1	2	1	2	4
1	p_5	2	2	2	4	1	p_5	2	2	2	4

Na podstawie tej tabeli otrzymujemy dwa ciągi działań palacza (p_0, p_1, p_2, p_3, p_4, p_1, p_5, p_e) oraz (p_0, p_1, p_3, p_6, p_4, p_1, p_5, p_e)

Mamy zatem ustalony zbiór reguł decyzyjnych $P=\{p_1, .. ,p_6\},p_0,p_e$.

1. Otrzymujemy następujące zbiory $S(p_i)$: $S(p_0)=\{p_1\}$, $S(p_1)=\{p_2,p_3,p_5\}$, $S(p_2)=\{p_3\}$, $S(p_3)=\{p_4,p_6\}$, $S(p_4)=\{p_1\}$, $S(p_5)=\{p_e\}$, $S(p_6)=\{p_4\}$.

2. Podział zbioru $P\cup\{p_e\}$ przez relację ρ

$P\cup\{p_e\}/\rho=[p_1]\cup[p_2,p_3,p_5]\cup[p_4,p_6]\cup[p_e]$.

3. Stąd $S(p_0),S(p_4)\in h_\rho^{-1}([p_1])$; $S(p_5)\in h_\rho^{-1}([p_e])$; $S(p_3),S(p_6)\in h_\rho^{-1}([p_4,p_6])$;

$S(p_1),S(p_2)\in h_\rho^{-1}([p_2,p_3,p_5])$.

4.Wyliczamy zbiory $F(p_i,p_j)$ i=0,1,2,..,6 j=1,2,..,6,e;

F	p_1	p_2	p_3	p_4	p_5	p_6	p_e
p_0	$P-\{p_1,p_4\}$	P	P	P	P	P	P
p_1	P	p_3	p	P	∅	P	P
p_2	P	P	p_4,p_1	P	P	P	P
p_3	P	P	P	∅	P	p_1,p_4	P
p_4	p_5	P	P	P	P	P	P
p_5	P	P	P	P	P	P	∅
p_6	P	P	P	∅	P	P	P

Wyliczamy estymatory funkcji ϕ i δ oraz zbiory $LV(p_i)$ dla każdego bloku podziału Π.

5. Bloki jednoelementowe $[p_1]$, $[p_e]$:

5.1. $LV_{[p_1]}=\{(p_1)\}$; $LV_{[p_e]}=\{(p_e)\}$.

5.4. $V(p_0)=(p_1)$; $V(p_4)=(p_1)$; $V(p_5)=(p_e)$.

5. Blok $[p_4,p_6]$:

5.2.1. $\overline{\Delta_{[p_4,p_6]}}(p_4)$={ wszystkie reguły }; $\overline{\Phi_{[p_4,p_6]}}(p_4)$={ wszystkie reguły };

$\overline{\Delta_{[p_4,p_6]}}(p_6)$={ p_4}; $\overline{\Phi_{[p_4,p_6]}}(p_6)$={ wszystkie reguły }.

5.2.2. $\Sigma=\begin{bmatrix} p_4 & p_6 \\ p_4 & p_6 \end{bmatrix}$; $\Phi=\begin{bmatrix} p_4 & p_6 \\ p_4 & p_6 \end{bmatrix}$.

5.2.3. $\Psi^{(0)}=\begin{bmatrix} p_4 & p_6 \\ p_4 & p_6 \end{bmatrix}$; $\Psi^{(1)}=\Psi^{(0)}\Phi=\begin{bmatrix} p_6p_4 & p_4p_6 \\ p_6p_4 & p_4p_6 \end{bmatrix}$.

5.2.4. $LV_{[p_4,p_6]}(p_4)$={ (p_6,p_4), (p_4,p_6) }; $LV_{[p_4,p_6]}(p_6)$={ (p_6,p_4) }.

5.3. $|LV_{[p_4,p_6]}(p_6)| \le |LV_{[p_4,p_6]}(p_4)|$.

5.4. $V(p_3)=(p_6,p_4)$, $V(p_6)=(p_4,p_6)$.

5. Blok [p_2,p_3,p_5]:

postępujemy analogicznie; ponieważ blok składa się z trzech elementów, zajdzie konieczność obliczania macierzy $\Psi^{(2)}$. Ostatecznie dla tego bloku otrzymamy następujące zbiory:

$LV_{[p_2,p_3,p_5]}(p_2)=\{ (p_2,p_3), (p_5,p_3) \}$; $LV_{[p_2,p_3,p_5]}(p_3)= LV_{[p_2,p_3,p_5]}(p_5)=\{ (p_3,p_2), (p_5,p_2),$ $(p_5,p_3,p_2), (p_3,p_5,p_2), (p_2,p_3), (p_5,p_2,p_3),(p_5,p_3), (p_2,p_5,p_3), (p_3,p_2,p_5), (p_3,p_5), (p_2,p_5)$ $(p_2,p_3,p_5) \}$.

5.3. $| LV_{[p_2,p_3,p_5]}(p_2) | \leq | LV_{[p_2,p_3,p_5]}(p_3) | = | LV_{[p_2,p_3,p_5]}(p_5) |$.

5.4. Wybieramy następujące porządki $V(p_2)=(p_2,p_3)$, $V(p_1)=(p_5,p_2,p_3)$.

7. Mając wybrane porządki dla wszystkich reguł, otrzymujemy wartości funkcji δ i ϕ i wywnioskowany automat.

	δ	ϕ
p_1 - startowa	p_5	λ
p_2	p_2	p_3
p_3	p_4	λ
p_4	p_1	p_6
p_5	λ	p_2
p_6	p_6	p_4

Zinterpretujmy nasze wyniki: proces podejmowania decyzyji rozpoczyna się od reguły startowej, następnie sprawdzamy, jakie wartości przyjęły atrybuty warunkowe i próbujemy zastosować regułę decyzyjną wskazaną przez funkcję δ. Jeżeli próba zakończyła się powodzeniem, to powtarzamy rozumowanie (tym razem dla reguły decyzyjnej ostatnio użytej), jeżeli próba zakończyła się fiaskiem, próbujemy zastosować regułę decyzyjną wskazaną przez funkcje ϕ. Jeżeli również tej reguły decyzyjnej nie można zastosować, przechodzimy dalej poprzez funkcję ϕ (od ostatniej reguły decyzyjnej, której nie można było zastosować). Proces taki trwa do momentu napotkania reguły stopu λ. Problem odmierzania czasu (odstępów sześciominutowych) pomijamy.

Otrzymawszy automat (P, δ, ϕ, p_s) możemy wypełnić dwie dodatkowe kolumny tablicy decyzyjnej, które będą zawierały wskazania dotyczące następnej reguły decyzyjnej do zastosowania. Otrzymamy wtedy również przegląd tego, w jaki sposób zmieniają się stany charakterystyczne sterowanego obiektu (w tym przypadku pieca obrotowego do wypalania klinkieru).

Graficznie wyprowadzany automat można również przedstawić w postaci grafu skierowanego, którego wierzchołki etykietowane są odpowiednimi wartościami atrybutów warunkowych, krawędzie etykietowane są wartościami odpowiednich atrybutów decyzyjnych, tzn. działaniami, jakie należy podjąć.

5. Wnioski

Przedstawiono propozycję podejścia do rozwiązania problemu przejmowania wiedzy i modelowania sposobu zachowania się człowieka, na podstawie przykładów jego działań, np. w procesie rozwiązywania zadań lub podejmowania decyzji.

Wykorzystując tablicę obserwacji i oparte na teorii zbiorów przybliżonych [8] metody ekstrakcji reguł decyzyjnych, można otrzymać odpowiednią regułową bazę wiedzy.

Do reprezentacji regułowej bazy wiedzy wykorzystano tablicę decyzyjną , którą docelowo rozszerzono o dwie dodatkowe kolumny (w części decyzyjnej) pozwalające na stworzenie pewnego rodzaju automatu pokazującego, którą regułę decyzyjną należy zastosować jako następną w przekształcaniu pewnych danych wejściowych.

Opierając się na pracy [8] podano algorytm wyprowadzający taki automat.

Ponieważ istnieje sporo pytań związanych zarówno z tą metodą (koszt obliczeń, w pewnych przypadkach problem niedeterministycznych reguł decyzyjnych), jak i z innymi metodami próbującymi wyprowadzać model zachowania się ze zbioru przykładów [11], wydaje się, że rozpatrzenie warunków, w szczególności celu, jaki będziemy chcieli osiągać za pomocą wyprowadzanego automatu, będzie miało duże znaczenie dla uproszczenia algorytmu jego syntezy.

LITERATURA

[1] Węgrzyn S.: Podstawy informatyki. PWN, Warszawa 1982.

[2] Zadeh L.A.: Soft Computing. Communications of the ACM, 1994, vol.37, nr 3.

[3] Kowalski R.: Logika w rozwiązywaniu zadań. WNT, Warszawa 1989.

[4] Mrózek A.: Modele wnioskowania operatorów i ich wykorzystanie w komputeryzacji obiektów technologicznych. ZN Polit.Śl., ser. Informatyka z.14, 1989.

[5] Michalski R.S., Mitchell T., Carbonell J.: Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann 1983, 1986, 1990.

[6] Płonka L.: Synteza systemu przetwarzania informacji opartego o regułową bazę wiedzy. Rozprawa doktorska, IITiS PAN, Gliwice 1996.

[7] Pawlak Z.: Decision Table Computer. Bulletin of the Polish Academy of Sciences, Technical Sciences, 1986, vol.34, nr 9-10.

[8] Pawlak Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht 1991.

[9] Witteveen C., Boelens H.: Inferring Control Structures from the Behaviour of a Production System. Information and Control 51, 1981.

[10] Coelho H., Cotta J.C.: Prolog by Example. Springer Verlag, Berlin 1991.

[11] Koskimies K., Makinen E.: Automatic Synthesis of State Machines from Trace Diagrams. Software-Practice and Experience, 1994, vol.24.

Recenzent: Prof. dr hab. Andrzej Skowron

Wpłynęło do Redakcji 8 września 1997 r.

Abstract

In this article we describe the problem of acquisition, representation and utilization with the aid of computer knowledge about „human" way of tasks solving.

We have established following way of representation the „human`s" algorithm:

<algorithm>=<rules base of knowledge>+<choice of decision rule mechanism>

Mathodes of gaining decision rules and synthesis of knowledge base were discussed in chapter 2.3.1. The component controlling the choice of decision rule specification was described by support of automaton idea as deterministic production system DPS (2.3.2).

We established the rules sequence from knowledge base as a protocol of man`s behaviour. Based on examples of human`s behaviour we gave the constracting DPS algorithm, modelling the theory given in [8].

In the chapter 3 we introduced several basic definitions suppored by which the mentioned algorithm was constructed.

The chapter 4 contains three examples of applying the introduced algorithm, among them the example of finding the way of stocker of rotary clinker klin in a cement plant behaviour (4.3).

It seems to be suitable in further developing of this method to make reach to attainments in domain of cognitive sciences. The introduction of goal notion defining tasks of inferring automaton may also turned out to be helpful.