ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996
Seria: INFORMATYKA z. 30 Nr kol. 1315

Jerzy KĘDZIERA
Leszek PŁONKA

ROUGH STEROWNIK JAKO ELEMENT INTELIGENTNEGO SYSTEMU ROZPROSZONEGO

Streszczenie. W artykule przedstawiono sposób realizacji układu Rough sterownika opartego na procesorze tablic decyzyjnych. Opisano konstrukcję, która wykorzystuje rozwiązania zastosowane podczas realizacji prototypu. Szczególną uwagę zwrócono na możliwość implementacji sieci Rough sterowników w rozproszonym inteligentnym systemie sterowania.

ROUGH CONTROLLER IN SMART DISTRIBUTED CONTROL SYSTEMS

Summary. The paper presents the implementation of a rough controller based on a processor of decision tables. It describes the design of the controller, utilizing the solutions applied in the realization of the prototype. Particular attention is paid to the possibility of implementing the rough controllers in smart distributed control systems.

COMMANDEUR "ROUGH" DANS UN SYSTEME INTELLIGENT DE COMANDE DISTRIBU

Résumé. L'article présente la manière de réalization d'un commandeur "Rough" basée sur un processeur de tableaux decisifs. Une construction qui met à profit les solutions appliquées pendant la réalization d'un prototype est décrite. Une attention essentielle est consacrée à la possibilité d'implementation de reseau de commandeurs "Rough" dans un système intelligent de commande distribu.

1. Wprowadzenie

Współczesne systemy sterowania zapewniają interakcje pomiędzy wieloma elementami wykonawczymi i pomiarowymi. Niesie to za sobą konieczność stosowania rozbudowanych sterowników posiadających wiele układów wejść i wyjść. Program obsługi takiego sterownika jest rozbudowany i złożony. W celu zapewnienia w takim układzie odpowiednich relacji czasowych konieczne jest zastosowanie rozbudowanych jednostek obliczeniowych. Realizacja układu sterowania jest kosztowna ze względu na stosowanie zaawansowanych technologicznie układów oraz konieczność zastosowania odpowiedniego rodzaju przewodów do elementów interakcji z obiektem. Taka realizacja powoduje zarazem zwiększone prawdopodobieństwo występowania uszkodzeń.

Alternatywnym rozwiązaniem jest dokonanie rozproszenia zasobów stosując sieć inteligentnych zintegrowanych sterowników. Idea ta polega na wykorzystaniu w systemie sterowania elementów integrujących element wykonawczy lub czujnik z układem akwizycji oraz łączem komunikacyjnym. Realizacja programu sterowania jest rozproszona pomiędzy poszczególne węzły sieci, z których każdy realizuje tylko fragment programu odpowiedni dla elementu wykonawczego lub czujnika odpowiadającemu danemu węzłowi sieci, przekazując do rozproszonego systemu sterowania tylko te dane, które są wykorzystywane przez inne węzły.

Węzeł sieci rozproszonego systemu sterowania może realizować algorytm akwizycji oparty na różnych metodach. Jedną z nich jest metoda zbiorów przybliżonych (ang. *Rougsel theory*), która jest podstawą realizacji Rough sterownika jako elementu rozproszonego systemu sterowania.

2. Budowa i funkcje Rough sterownika

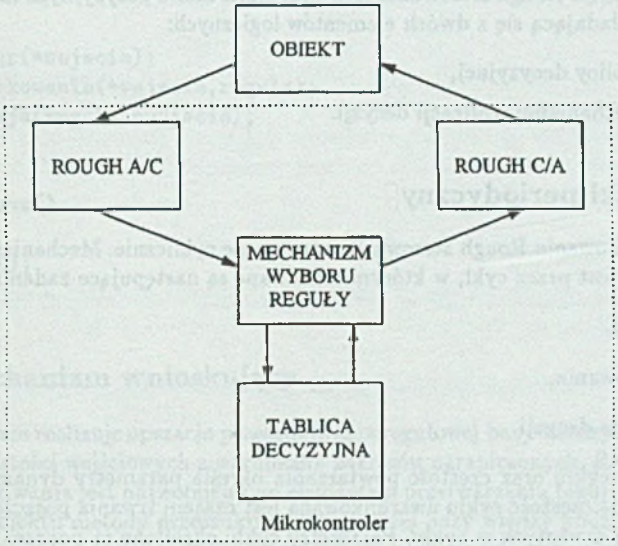
Idea Rough sterownika pojawiła się w [1], a jego implementacja oparta została na następujących elementach wykonawczych:

- zbiór reguł decyzyjnych,
- przetwornik A/C o ustalonej przez atrybuty nieliniowości - Rough A/C,
- przetwornik C/A o ustalonej przez reguły nieliniowości - Rough C/A,
- układ akwizycji pomiarów i reguł.

Schemat Rough sterownika przedstawia rysunek 1.

2.1. Regułowa baza wiedzy

Zbiór reguł decyzyjnych implementowany w Rough sterowniku, nazywany regułową bazą wiedzy, reprezentowany jest w postaci tablic decyzyjnych, która posiada następujące pola:



Rys. 1. Schemat Rough sterownika
Fig. 1. Rough Controller configuration

- pole warunków,
- pole decyzji.

Zbiór reguł decyzyjnych określony jest przez instrukcje warunkowe. Odwołanie do reguły oraz odpowiadających decyzji następuje przez sprawdzenie warunków dla atrybutu wejściowego:

IF warunki THEN decyzje.

2.2. Rough sterownik jako procesor tablic decyzyjnych

W celu realizacji Rough sterownika posłużono się koncepcją procesora tablic decyzyjnych. Zastosowanie takiej analogii wynika ze sposobu realizacji dostępu do reguł decyzyjnych. Regułowa baza wiedzy Rough sterownika jest przypadkiem szczególnym tablicy decyzyjnej o następujących uwarunkowaniach:

1. Regułowa baza wiedzy określona jest za pomocą warunków zakresów ograniczonych:
   (a) nie nakładających się – dla argumentu jednowymiarowego lub wielowymiarowego w przypadku szczególnym,
   (b) nakładających się – dla argumentu wielowymiarowego w przypadku ogólnym.

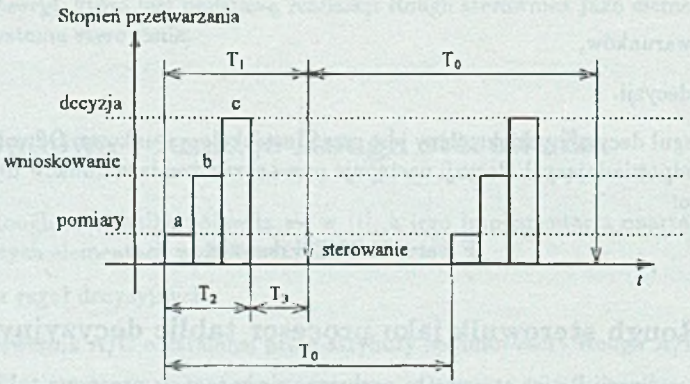
2. Konstrukcja Rough sterownika jako procesora tablic decyzyjnych narzuca konfigurację składającą się z dwóch elementów logicznych:
   (a) tablicy decyzyjnej,
   (b) mechanizmu realizacji decyzji.

2.3. Cykl periodyczny

Proces sterowania Rough sterownika odbywa się cyklicznie. Mechanizm realizacji decyzji opisany jest przez cykl, w którym relizowane są następujące zadania:

- pomiary,
- wnioskowanie,
- realizacja decyzji.

Czas trwania cyklu oraz częstość powtarzania określa parametry dynamiczne sterowania. Minimalna częstość cyklu uwarunkowana jest czasem trwania poszczególnych zadań przetwarzania zawartych w cyklu. Parametry cyklu określone są poprzez parametry dynamiczne obiektu. Schemat ilustrujący ideę cyklu periodycznego przedstawiony jest na rys. 2.



Rys. 2. Ilustracja cyklu periodycznego
Fig. 2. periodic cycle

Na początku cyklu realizowane są pomiary, przekazując do układu wnioskowania wartości cech obiektu. Na podstawie danych wejściowych określana jest reguła zawarta w regułowej bazie wiedzy przez warunki zakresów ograniczonych. Określonej regule odpowiada decyzja przetworzona w układzie decyzyjnym na odpowiadające jej wartości wyjściowe. Realizacja sterowania opisana jest następująco:

```
cykl()
{
        pomiar(*wejscie);
        wnioskowanie(*wejscie,regula);
        decyzja(regula,*wyjscie);
}

interrupt timer()
{
        cykl();
}
```

2.4. Mechanizm wnioskujący

Wnioskowanie realizuje operacje przeszukiwania regułowej bazy wiedzy na podstawie porównania wartości wejściowych z warunkami zakresów ograniczonych. Realizacja algorytmu przeszukiwania jest najwolniejszym elementem przetwarzania reguł. W zależności od wymagań obiektu metody przeszukiwania regułowej bazy wiedzy podzielono na:

1. Liniowe – polegające na kolejnym porównywaniu warunków ograniczonych według kolejności w regułowej bazie wiedzy:
   - prosty algorytm przeszukiwania,
   - brak konieczności analizy wydajnościowej regułowej bazy wiedzy,
   - długi czas przeszukiwania;
2. Maskowanie – polegające na przypisaniu każdej możliwej wartości wejściowej odpowiadającej wartości wyjściowej:
   - krótki czas realizacji,
   - konieczność przetworzenia bazy wiedzy,
   - duża zajętość pamięci;
3. Mieszające – wykorzystujące funkcje mieszające (ang. *hashing*);
4. Optymalizowane pod względem:
   - czasowym,
   - zajętości pamięci.

Przeszukiwanie liniowe jest najprostszym lecz mało wydajnym algorytmem. Poprawienie wydajności można osiągnąć poprzez analizę wydajnościową oraz przetworzenie regulowej bazy wiedzy. Jedną z metod jest sortowanie, polegające na ustaleniu kolejności realizacji poszczególnych porównań [2].

3. Implementacja Rough sterownika

W Instytucie Informatyki Teoretycznej i Stosowanej PAN w Gliwicach został zrealizowany prototyp Rough sterownika. Jego konstrukcja oparta została na mikrokontrolerze *Motorola MC68HC16* [4]. Prototyp realizuje dostęp do elementu bazy wiedzy według algorytmu przeszukiwania opartego na sortowaniu tablicy decyzyjnej. Rough sterownik posiada następujące cechy:

- dwa wejścia analogowe o rozdzielczości 10 bitów,
- łącze sieciowe pozwalające na pracę sterownika w rozproszonym inteligentnym systemie sterowania,
- jedno wyjście analogowe lub 16-bitowe wyjście cyfrowe.

Regułowa baza wiedzy Rough sterownika zapisana jest w postaci tablicy w pamięci RAM sterownika, co umożliwia dokonywanie zdalnej modyfikacji reguł.

3.1. Praca sterownika w sieci

Zrealizowany Rough sterownik może pracować autonomicznie lub w inteligentnej sieci sterowników. Różnica polega tylko na odpowiednim skonfigurowaniu mikrokontrolera. W trybie sieciowym można wprowadzić dodatkowe wejście oraz wyjście danych obrazujące stan systemu sieciowego. W trybie autonomicznym rozwiązanie to znajduje zastosowanie przy zdalnym monitorowaniu i symulacji elementów wejściowych i wyjściowych.

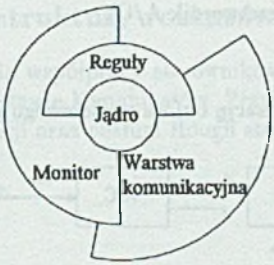
Zaimplementowany protokół komunikacyjny pozwala na dokonywanie zmian parametrów przetwarzania, jak również służy do wymiany informacji pomiędzy poszczególnymi węzłami sieci.

3.2. Struktura oprogramowania Rough sterownika

Program realizujący przetwarzanie tablic decyzyjnych został napisany z wykorzystaniem kompilatora SmallC. Elementem realizującym i monitorującym pracę Rough sterownika jest *oprogramowanie bazowe* o strukturze pokazanej na rysunku 3.

Struktura bazowa pełni funkcję systemu operacyjnego Rough sterownika z obsługą pracy sieciowej i składa się z następujących elementów:

- jądro – realizujące podstawowe funkcje systemu takie, jak:
  - — inicjalizacja układu,
  - — obsługa przerwań sprzętowych i programowych,
  - — zarządzanie procesami,
- reguły – będące procesem realizującym interpretację reguł decyzyjnych,



Rys. 3. Schemat systemu operacyjnego Rough sterownika
Fig. 3. Rough Controller operating system

- monitor – odpowiedzialny za obsługę systemu sieciowego oraz synchronizację procesów,
- warstwa komunikacyjna – realizująca wymianę danych pomiędzy sterownikiem a otoczeniem.

Jądro systemu operacyjnego realizuje funkcje wymiany informacji pomiędzy systemem sterownika (przerwania sprzętowe i programowe) i procesami wyższych warstw. Przerwaniami sprzętowymi zaimplementowanymi w układzie Rough sterownika są:

- przerwanie licznikowe,
- przerwanie od układu nadajnika i odbiornika.

Przerwanie licznikowe służy do kontroli zależności czasowych w systemie, a przerwanie od układu nadawczo-odbiorczego wykorzystywane jest w części monitora.
Realizacja tablic decyzyjnych jest procesem cyklicznym inicjowanym przez zdarzenie.

3.3. Rozproszona tablica decyzyjna

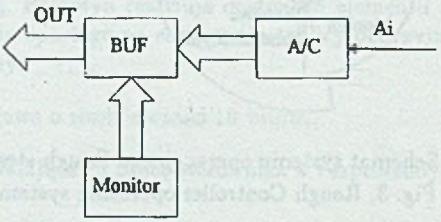
W sieci Rough sterowników regułowa baza wiedzy w postaci tablicy decyzyjnej ulega rozproszeniu [3]. Ze względu na konieczność wykorzystania przez pojedynczy sterownik nie tylko lokalnych zasobów, lecz również informacji pochodzących od innych sterowników, wykorzystano pojęcie stanu systemu sterowania [3]. Do zmiany stanu systemu dochodzi tylko w momencie, kiedy jakakolwiek wielkość mierzona zmieni swoją wartość w zakresie istotnym dla sterowania. Decyzję o zmianie stanu podejmuje ten sterownik, w którym dana wielkość jest mierzona. Informacja zmiany stanu jest rozsyłana w sieci globalnie lub tylko dla tych węzłów, które właśnie ten stan określa.

3.4. Układy wejścia–wyjścia

W Rough sterowniku istnieją dwa źródła sygnałów wejściowych:

- analogowy układ pomiarowy (przetwornik A/C),
- wartość stanu systemu z sieci.

W tym celu zaproponowano organizację bufora wejściowego jak pokazuje rysunek 4.



Rys. 4. Realizacja bufora wejściowego
Fig. 4. realization of the input buffer

Wejście monitora przekazuje do bufora (BUF) odczytaną z sieci wartość stanu systemu. Niezależnie od tego wartości mierzone przez przetwornik A/C (A_i) są wpisywane na kolejne pozycje wektora wejściowego w buforze. Cały wektor wejściowy dostępny jest w systemie sterownika przez strumień (OUT).

Zmiana zawartości bufora występuje w momencie:

- wpisu danej z układu Monitora ,
- na żądanie systemu wczytania danej z przetwornika A/C podczas realizacji cyklu periodycznego.

Uaktualnienie danej w buforze powoduje realizację przetwarzania reguł.

Po dokonaniu przetwarzania decyzja umieszczana jest w buforze wyjściowym, do którego zostaje przekazana jako sygnał sterowania. Drugie pole wektora bufora wyjściowego stanowi obraz stanu systemu i w przypadku zmiany dokonywane jest rozesłanie danej w sieci sterowników.

4. Konfigurowanie układu Rough sterownika

Konfiguracja układu Rough Controller'a odbywa się w dwóch etapach:

- inicjalizacja układu mikrokontrolera,
- konfiguracja z systemu nadrzędnego z udziałem Monitora.

Pierwszy etap odpowiedzialny jest za stworzenie wymaganego środowiska w celu realizacji układu tablic decyzyjnych. Za pomocą Monitora konfigurowany jest układ wejść, wyjść oraz ładowane są reguły decyzyjne. Poszczególne rozkazy konfigurujące Monitora opisane są poniżej.

4.1. Definicje struktury rozkazów

W celu umożliwienia współpracy sterowników w sieci opracowano protokół komunikacyjny oparty na zestawie komunikatów. Przyjęto następujący format komunikatów sieciowych do konfiguracji oraz obsługi Rough sterownika:

- Rozkazy sterujące:

```
typedef struct
{
    byte    n_inputs;   //liczba wejsc sterownika
    byte    n_outputs   //liczba wyjsc sterownika
} SetStructure          //kod ramki wynosi 1


typedef struct
{
} Reset                 //kod ramki wynosi 2


typedef struct
{
    word    rule[]      //reguly 2*(n_inputs+n_outputs)
} SetRule               //kod ramki wynosi 3


typedef struct
{
    word    values[]    //wartosci wejsc sterownika [n_inputs]

} SetInputs             //kod ramki wynosi 4


typedef struct
{
} GetOutputs            //kod ramki wynosi 5


typedef struct
{
    word    value       //wartosc sterowania nieokreslonego warunku
} Otherwise             //kod ramki wynosi 6
```

- Informacje zwrotne:

```
typedef struct
{
    byte    errcode     //potwierdzenie odbioru komunikatu OK|ERROR
} Confirm               //kod ramki wynosi 129


typedef struct
{
    word    values[]    //Wartosci sterowania [n_outputs]
} Outputs               //kod ramki wynosi 130
```

Powyższy zestaw rozkazów został zaimplementowany i przetestowany w konfiguracji jednego sterownika i systemu nadrzędnego. Dla sieci sterowników modyfikacje nie są konieczne. Zestaw rozkazów można rozbudowywać. Przyjęto konwencję, że rozkazy sterujące zajmują przestrzeń kodów od 1 do 127, informacje zwrotne od 129 do 255, a pozostałe wartości są zarezerwowane.

5. Podsumowanie

Konstrukcja Rough sterownika jest w pełni przydatna do realizacji inteligentnego rozproszonego systemu sterowania. Fizycznie istniejący prototyp Rough sterownika został przetestowany do współpracy z tylko jednym dodatkowym węzłem sieci, spełniającym rolę nadzorcy. Projekt został jednak tak przygotowany, że współpraca większej ilości Rough sterowników w sieci nie wymaga tworzenia nowej konstrukcji. Metody rozproszenia bazy wiedzy w sieci nie są w pełni opracowane i testy na rzeczywistych obiektach sieci sterowników wymagają jeszcze przygotowania.

LITERATURA

[1] Mrózek, A., Płonka, L.: *Rough Sets for Controller Synthesis.* Third International Workshop on Rough Sets and Soft Computing, San Jose, CA, November, 1994.

[2] Pollack, S.L., Hicks, T.H., Jr., Harrison, W.J.: „Tablice decyzyjne". PWN, Warszawa, 1975.

[3] Płonka, L., Mrózek, A., Winiarczyk, R., Maitan, J.: *Implementing Rule-Oriented Knowledge Bases on Smart Networks.* Fourth International Workshop on Intelligent Information Systems, Augustów, June, 1995.

[4] Motorola: *MCU16 Reference Manual*, 1992.

Recenzent: Prof. dr hab. inż. Andrzej Grzywak

Wpłynęło do Redakcji 19 grudnia 1995 r.

Abstract

Modern control systems require interaction between numerous sensors and actuators, hence the need for controllers equipped with complex I/O subsystems. Software running on such controllers may be very complex. Ensuring the timing requirements may require powerful processing units. The implementation of such control systems may be expensive, also because of the high cost of the wiring required for proper interaction with the controlled plant. This also increases the probability of failure.

An alternative solution is a network of smart, integrated controllers. The idea is to integrate a sensor or actuator with an embedded microcontroller and a communication network. In smart distributed systems, the execution of the control program is distributed among many network nodes. Each node is responsible for a small fragment of the control program and sends to the network only those data items that are utilized by other nodes.

The smart network nodes may execute a control algorithm obtained with various methods. One such method is based on rough set theory, that is the inspiration for an implementation of the rough controller as an element of a distributed control system.