

Metoda wyznaczania bezkolizyjnych dróg przesyłania danych w systemie o strukturze nadmiarowej

Jan CHUDZIKIEWICZ

Zakład Systemów Komputerowych, Instytut Teleinformatyki i Automatyki WAT,
ul. Kaliskiego 2, 00-908 Warszawa

STRESZCZENIE: W niniejszym artykule zaprezentowano metodę oraz bazujący na tej metodzie algorytm wyznaczania bezkolizyjnych ścieżek przesyłania danych w systemie o strukturze nadmiarowej. Przykładem takiej struktury jest struktura typu hipersześcianu. Dla opracowanej metody przyjęto założenie, że każdy z węzłów przechowuje informacje o stanie niezawodnościowym całego systemu. Ponadto w artykule zaprezentowano, opracowany na potrzeby implementacji algorytmu w systemie Windows®, sterownik pakietowy. Omówiono także mechanizmy wbudowane w system, które ułatwiają implementację opracowanego algorytmu.

SŁOWA KLUCZOWE: hipersześcian, samodiagnozowanie, systemy tolerujące błędy

1. Wprowadzenie

Problematyka opracowywania metod zapewniających odpowiednią efektywność transmisji danych w systemach wielokomputerowych o nadmiarowej strukturze, jest szeroko omawiana w literaturze światowej. Podobny oddźwięk znajdują zagadnienia związane z możliwością rekonfiguracji struktury takiego systemu tak, aby po zaistnieniu awarii graf opisujący tę strukturę był spójny. Mając powyższe na uwadze, w niniejszym artykule przedstawiono metodę wyznaczania bezkolizyjnych dróg przesyłania danych w strukturach typu hipersześcianu [10] [3].

Systemy o strukturze logicznej n -wymiarowego hipersześcianu należą do klasy systemów tolerujących błędy i charakteryzują się dużą złożonością dla $n > 3$ [10][5][4][3].

Problem przesyłania danych w systemach o strukturze hipersześcianu jest szeroko analizowany w literaturze przedmiotu. Między innymi Gordon i Stout przedstawiają metodę nazwaną przez nich „sidetracking” [8]. Dla metody „sidetracking” przyjęto założenie, że w każdym z węzłów przechowywane są informacje o stanie niezawodnościowym węzłów sąsiednich. Informacja z danego węzła jest przesyłana przez losowo wybraną ścieżkę, incydentną z węzłem będącym w stanie zdadności. Jeżeli dany węzeł nie sąsiaduje z węzłami zdadnymi, informacja jest blokowana i przesyłana z powrotem do węzła, z którego pierwotnie była wysłana. Metoda ta, ze względu na sposób wyboru ścieżki transmisji informacji, charakteryzuje się małym prawdopodobieństwem przesłania informacji dla określonej liczby niezdatnych węzłów w systemie oraz dużym opóźnieniem czasowym. Inna metoda zaproponowana przez Chena nosi nazwę „backtracking” [1]. Dla metody „backtracking” przyjęto założenie, że informacje o węzłach pośredniczących w przesyłaniu pakietu są dodawane do niego. W przypadku, gdy dane dotrą do węzła, który sąsiaduje z węzłami niezdatnymi informacja ta jest wykorzystywana do zwrotnego przesłania danych do węzła wcześniejszego. Wadą tego rozwiązania jest wprowadzanie do przesyłanych danych nadmiarowej informacji oraz duże opóźnienia czasowe.

W niniejszym artykule zaprezentowano metodę wyznaczania bezkolizyjnych ścieżek przesyłania danych w systemie o strukturze hipersześcianu, bazującą na wyznaczeniu pokrycia grafu opisującego daną strukturę. Dla opracowanej metody przyjęto założenie, że każdy z węzłów przechowuje informacje o stanie niezawodnościowym całego systemu. Ponadto w artykule zaprezentowano, opracowany na potrzeby implementacji algorytmu w systemie Windows®, sterownik pakietowy. Omówiono także mechanizmy wbudowane w system, które ułatwiają implementację opracowanego algorytmu.

2. Pojęcia podstawowe

Niech Z^n oznacza zbiór n -wymiarowych wektorów binarnych.

Oznaczmy:

$$(s_1, \dots, s_n) = \{z \in Z^n : ((s_i \neq x) \Rightarrow (z_i = s_i)) \wedge ((s_i = x) \Rightarrow (z_i \in \{0, 1\})) \mid (s_i \in \{0, 1, x\}, 1 \leq i \leq n),$$

gdzie: x oznacza wartość nieokreśloną (0 lub 1).

$Z(s)$ -zbiór podsześcianów 0-wymiarowych (zbiór wektorów

$z = (z_1, \dots, z_n)$, $(z_i \in \{0, 1\}, 1 \leq i \leq n)$ podsześcianu s ($s \in S^n$)).

Określenie 1. n -wymiarowym hipersześcianem binarnym nazywamy graf

zwykły $G' (G' = \langle E, U' \rangle, |E| = 2^n, |U'| = n \cdot 2^{n-1})$ o 2^n węzłach, z których każdy opisany jest odpowiednim wektorem binarnym $z (z = (z_1, \dots, z_n), z_i \in \{0, 1\}, 1 \leq i \leq n, z \in Z^n, |Z^n| = 2^n)$ oraz o $n \cdot 2^{n-1}$ krawędziach łączących te węzły, których opisujące je wektory odległe są o 1 według miary Hamminga.

Strukturę n -wymiarowego hipersześcianu binarnego będziemy dalej oznaczać przez H^n a graf częściowy tej struktury przez H_t^n . Indeks t oznacza liczbę krawędzi grafu struktury H_t^n .

Dalej węzły grafu H^n będą reprezentować komputery (procesory), a jego krawędzie - linie transmisji danych między tymi komputerami (procesorami), które są incydentne z określoną krawędzią.

Określenie 2. Łańcuchem τ o długości $k (0 \leq k \leq 2^n)$ w H^n nazywamy spójny podgraf grafu H^n zawierający $k + 1$ węzłów, z których tylko dwa są stopnia pierwszego.

Węzeł stopnia pierwszego łańcucha nazywamy biegunem tego łańcucha.

Niech $Z(\tau)$ oraz $B(\tau) (B(\tau) \subseteq Z(\tau))$ oznaczają odpowiednio zbiór węzłów oraz biegunów łańcucha τ .

Łańcuch τ będziemy przedstawiać zarówno w postaci podgrafu $\langle Z(\tau) \rangle_{H^n}$ jak i w postaci zbioru $S(\tau)$ podsześcianów 1-wymiarowych $s (s \in S_1^n)$ takiego, że:

$$[s \in S(\tau)] \Leftrightarrow [\exists z', z'' \in Z(\tau) : z' + z'' = s].$$

Określenie 3. Mówimy, że łańcuchy τ' i τ'' w H^n są silnie wzajemnie niezależne, jeżeli $Z(\tau') \cap Z(\tau'') = \emptyset$.

Określenie 4. Mówimy, że zbiór $P (P = \{\tau_1, \dots, \tau_p\}, 1 \leq p \leq 2^{n-1})$ silnie wzajemnie niezależnych łańcuchów τ_1, \dots, τ_p , jest pokryciem H^n , jeżeli $\{Z(\tau_i) : 1 \leq i \leq p\} = Z^n$.

Określenie 5. Mówimy, że pokrycie $P (P \in \mathbf{P}_n)$ jest pokryciem klasy $\mathbf{K}(\lambda_1^{\delta_1}, \dots, \lambda_a^{\delta_a}) (\mathbf{K}(\lambda_1^{\delta_1}, \dots, \lambda_a^{\delta_a}) \in \mathbf{K}_n)$, jeżeli spełnione są następujące warunki:

$$(|S(\tau_i)| = \lambda_i) \wedge (\delta(b'(\tau_i), b''(\tau_i)) = \delta_i, \tau_i \in P, b'(\tau_i), b''(\tau_i) \in B(P), \lambda_i \in \lambda, i = \{1, \dots, |P|\})$$

gdzie:

\mathbf{K}_n - zbiór klas hipersześcianu H^n ,

\mathbf{P}_n - zbiór pokryć H^n ,

$\Lambda^a(b)$ ($2 \leq a \leq b$) - zbiór addytywnych a -członowych podziałów $\lambda = (\lambda_1, \dots, \lambda_a)$ ($\lambda \in \Lambda^a(b)$, $\lambda_i > 0$, $1 \leq i \leq a$, $\lambda_j \geq \lambda_{j+1}$, $1 \leq j \leq a-1$) liczby wierzchołków b hipersześcianu H^n .

Odległość Hamminga między dwoma wektorami binarnymi $b'(\tau_i)$ i $b''(\tau_i)$, będącymi biegunami łańcucha τ_i , spełnia zależność:

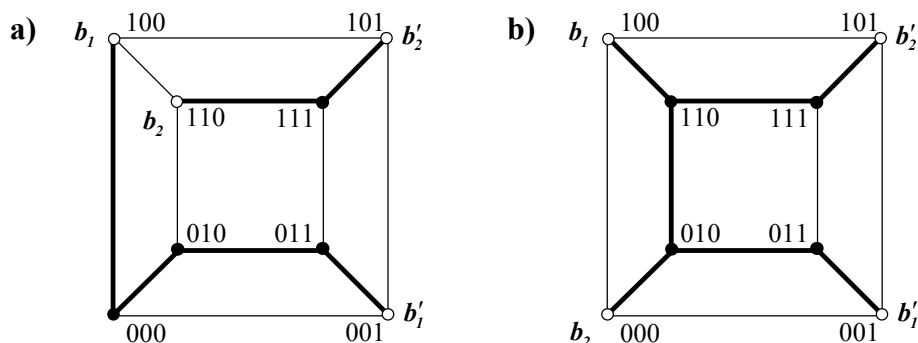
$$\delta(b'(\tau_i), b''(\tau_i)) = \sum_{k \in \{1, \dots, n\}} (b'(\tau_i)_k \oplus b''(\tau_i)_k),$$

gdzie: $b(\tau_i)_k$ - k -ta składowa wektora binarnego $b(\tau_i)$,

\oplus - suma modulo 2.

3. Metoda wyznaczania bezkolizyjnych ścieżek przepływu danych w systemie o strukturze nadmiarowej

Metoda wyznaczania bezkolizyjnych ścieżek polega na wyznaczeniu zbioru możliwych pokryć struktury H^n silnie wzajemnie niezależnymi łańcuchami prostymi. Efektem końcowym działania metody jest wyznaczenie wszystkich ścieżek pomiędzy elementami, które w danym momencie zamierzają realizować wymianę danych tak, aby nawzajem nie zakłócały one transmisji. Dla zobrazowania działania metody rozpatrzmy przykład wyznaczenia ścieżek pomiędzy dwiema parami elementów w strukturze H^3 . Postać jednego z możliwych pokryć pokazano na rysunku 1a.



Rys. 1. Przykład pokrycia $|P|=2$ struktury H^3

W przedstawionym przykładzie zostało wyznaczone pokrycie, dwoma

łańcuchami prostymi grafu struktury H^3 , umożliwiające bezkolizyjną wymianę danych pomiędzy węzłami (b_1, b'_1) i (b_2, b'_2) , dla których $\delta(b_1(\tau_1), b'_1(\tau_1))=2$, $|S(\tau_1)|=4$ oraz $\delta(b_2(\tau_2), b'_2(\tau_2))=2$, $|S(\tau_2)|=2$. Wyznaczone pokrycie należy do klasy $\mathbf{K}(4^2, 4^2)$.

Natomiast na rysunku 1b przedstawiono przypadek, w którym dla dwóch par węzłów (b_1, b'_1) i (b_2, b'_2) metoda nie będzie w stanie wyznaczyć pokrycia, gdyż takie pokrycie łańcuchami rozłącznymi nie istnieje. Przyjmuje się wówczas, że niektóre linie komunikacyjne (np. o największej przepustowości) będą wspólne dla obydwu par węzłów.

4. Algorytm wyznaczania bezkolizyjnych ścieżek przepływu danych w systemie o strukturze nadmiarowej

Wyznaczenie klas pokryć jest zadaniem złożonym, szczególnie dla $n \geq 4$, dlatego też należy w tym celu wspomóc się komputerem. Poniżej przedstawiono algorytm wyznaczania pokryć.

Oznaczenia:

$L(z', z'')$ - zbiór łańcuchów łączących węzły z' oraz z'' ,

$Z(\tau_j)$ - zbiór węzłów tworzących łańcuch τ_j ,

P - aktualnie wyznaczane pokrycie.

Algorytm rozpoczynamy od $i = 1$, czyli pierwszego łańcucha pokrycia.

Krok 1.

Wybierz, jako biegun początkowy łańcucha τ_j , nie wybrany węzeł $z' \in Z^n \setminus Z(\tau_j)$, $\tau_j \in P$, $j = \{1, \dots, i-1\}$, $i > 1$ o najmniejszej etykiecie.

Określ zbiór Z_i , ($Z_i = \{z'' \in Z^n \setminus Z(\tau_j) \cup z' : (\delta(z', z'')) = \delta_i\}$), $\tau_j \in P$, $j = \{1, \dots, i-1\}$, $i > 1$) biegunów końcowych łańcucha τ_j .

Krok 2.

Wyznacz zbiór $L(z', z'')$ ($L(z', z'') = \{\tau : |S(\tau)| = \lambda_i\}$, $\forall z'' \in Z_i$) łańcuchów łączących węzły z' oraz z'' tak, aby: ($\forall \tau_j \in P : Z(\tau_j) \cap Z(\tau) = \emptyset$, $j = \{1, \dots, i-1\}$, $i > 2$).

Krok 3.

Pobierz łańcuch τ ze zbioru łańcuchów dostępnych $L(z', z'')$ i dodaj go do pokrycia P . $L(z', z'') = L(z', z'') \setminus \tau$.

Jeżeli $i = a$, to przejdź do kroku 4 w przeciwnym przypadku $i = i + 1$ i przejdź do kroku 1.

Krok 4.

Sprawdź, czy nie wyznaczono wcześniej pokrycia odpowiadającego pokryciu P .

$$P \in K(\lambda_1^{\delta_1}, \dots, \lambda_a^{\delta_a}) \Leftrightarrow \neg \exists P' \in K(\lambda_1^{\delta_1}, \dots, \lambda_a^{\delta_a}) : \\ : (B(\tau_l) \neq B(\tau_k)) \wedge (Z(\tau_l) \neq Z(\tau_k)), \\ \tau_l \in P', \tau_k \in P, l, k = \{1, \dots, a\}.$$

Jeżeli nie wyznaczono wcześniej pokrycia odpowiadającego pokryciu P , to dodaj P do zbioru P_n . W przeciwnym przypadku odrzuć P . Przejdź do kroku 5.

Krok 5.

Czy $L(z', z'') = \emptyset$. Jeżeli nie, to przejdź do kroku 3. W przeciwnym przypadku, gdy $i > 1$, $i = i - 1$ powtórz krok 5. Jeżeli $i = 1$, sprawdź, czy węzeł, który nie był jeszcze biegunem ma etykietę mniejszą od 2^n . Jeżeli TAK, to przejdź do kroku 1. Jeżeli NIE, to przejdź do kroku 6.

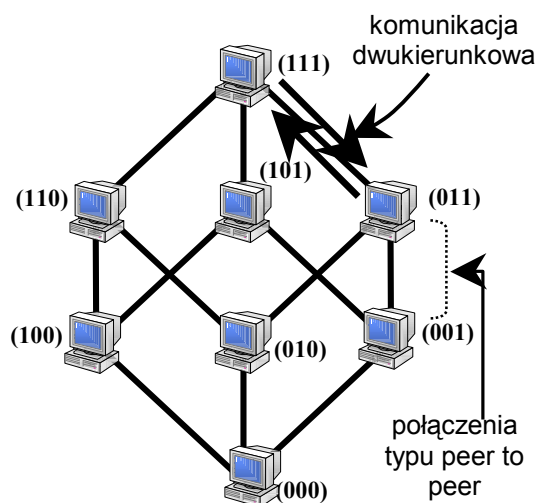
Krok 6.

Koniec działania algorytmu.

5. Implementacja w systemie Windows®

Sieć komputerowa o strukturze hipersześcianu jest zaliczana do sieci typu punkt-punkt (rysunek 2), co powoduje, że komunikacja pomiędzy komputerami, których odległość Hamminga jest większa niż 1, wymaga pośrednictwa innych komputerów.

Rozwiązanie takie wymaga zastosowania komputerów wyposażonych w n -interfejsów sieciowych, gdzie n jest wymiarem struktury hipersześcianu. Aby zapewnić możliwość komunikacji z wykorzystaniem wszystkich interfejsów, należało opracować odpowiedni sterownik i włączyć go w architekturę sieciową systemu Windows® (rysunek 3).



Rys. 2. Rozwiązanie komunikacji w sieci komputerowej o strukturze hipersześcianu

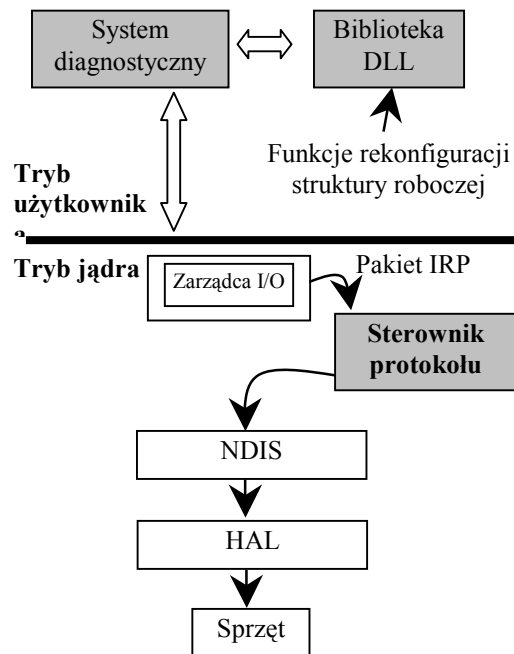
Architektura systemu Windows zakłada wykorzystanie dwóch poziomów uprzywilejowania, jak przedstawiono to na rysunku 3: trybu użytkownika oraz trybu jądra [7] [11].

W trybie użytkownika uruchamiane są aplikacje użytkowe. W naszym przypadku jest to system diagnostyczny nadzorujący poprawność działania całej struktury.

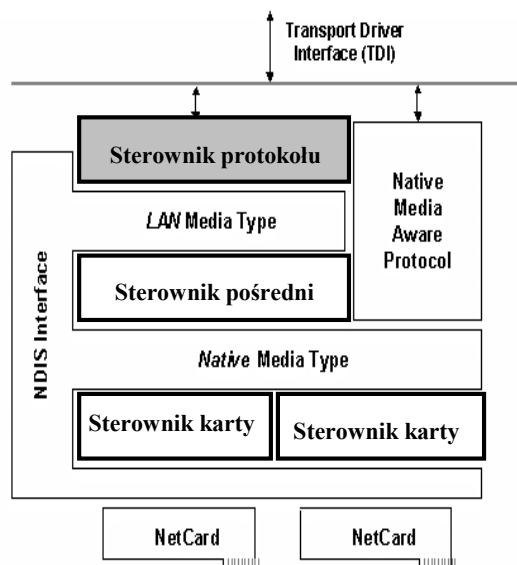
Specyfikacja interfejsu sterowników sieci (ang. NDIS - Network Driver Interface Specification) jest zrealizowana w Windows® jako biblioteka, która definiuje interfejsy między sterownikami poszczególnych warstw. Takie rozwiązanie pozwala na oddzielenie sterowników sprzętowych (niskiego poziomu) od sterowników wyższych warstw takich jak warstwa transportowa. NDIS przechowuje również informacje na temat stanu i parametrów sterowników sieciowych w tym wskaźników do funkcji, uchwytów i innych wartości.

NDIS wyróżnia następujące typy sterowników (patrz rysunek 4):

- sterownik miniportu (ang. Miniport driver);
- sterownik pośredni (ang. Intermediate driver);
- sterownik protokołu (ang. Protocol driver).



Rys. 3. Architektura systemu Windows klasy NT



Rys. 4. Typy sterowników NDIS

Sterownik miniportu (sterownik karty sieciowej) ma dwie podstawowe funkcje:

- zarządzanie kartą sieciową w tym wysyłanie i odbieranie danych;
- współpraca ze sterownikami wyższego poziomu, takimi jak sterowniki pośrednie czy sterowniki protokołu.

Sterownik miniportu komunikuje się z kartą sieciową i sterownikami wyższego poziomu poprzez funkcje z biblioteki NDIS. Biblioteka ta eksportuje zbiór funkcji (nazwy funkcji mają przedrostek Ndis), który zawiera wszystkie funkcje systemu operacyjnego wymagane przez sterownik karty sieciowej. Niemniej istotnym jest to, że sterownik karty sieciowej powinien udostępniać oddzielny zbiór funkcji przeznaczony dla biblioteki NDIS (nazwy funkcji mają przedrostek Miniport).

Współpraca sterownika miniportu z biblioteką NDIS i sterownikiem wyższego poziomu realizowana jest zgodnie z następującymi zasadami:

- Wysłanie pakietu przez sterownik protokołu powoduje wywołanie funkcji wyeksportowanej przez bibliotekę NDIS. NDIS przekazuje otrzymany pakiet do sterownika miniportu wywołując odpowiednią funkcję wyeksportowaną przez sterownik miniportu. Sterownik miniportu, będący ostatnim elementem na drodze przesyłania pakietu, przekazuje go do karty sieciowej. Operacja przekazania pakietu pomiędzy sterownikiem miniportu a kartą sieciową, realizowana jest poprzez wywołanie odpowiedniej funkcji Ndis przez sterownik.
- Odebranie przez kartę sieciową pakietu do niej zaadresowanego powoduje wysłanie komunikatu o tym zdarzeniu do sterownika miniportu. Sterownik miniportu korzystając z funkcji biblioteki NDIS wymusza przesłanie odebranego pakietu z karty sieciowej do pamięci podręcznej sterownika miniportu. Kolejną czynnością jest wywołanie przez sterownik miniportu odpowiedniej funkcji Ndis, dzięki której sterownik wyższego poziomu zostanie poinformowany o otrzymanym pakiecie.

Sterownik pośredni stosowany jest w hierarchii pomiędzy sterownikiem miniportu a sterownikiem protokołu, dlatego komunikować musi się z dwoma sterownikami:

- Od strony niższej warstwy sterownik pośredni świadczy usługi sterownika protokołu poprzez wyeksportowanie funkcji, której nazwa ma przedrostek Protocol. Funkcje te używane są przez NDIS do komunikacji, gdy sterownik miniportu przesyła informacje w górę stosu protokołów. Dla sterownika miniportu sterownik pośredni wygląda jak sterownik protokołu.
- Od strony wyższej warstwy sterownik pośredni świadczy usługi sterownika miniportu poprzez wyeksportowanie funkcji Miniport,

których używają sterowniki protokołu do komunikowania się z kartą sieciową. Dla sterownika protokołu sterownik pośredni wygląda jak sterownik miniportu.

Pomimo tego, że sterownik pośredni eksportuje zestaw funkcji Miniport dla wyższych sterowników stosu sieciowego, nie zarządza on fizycznie kartą sieciową, tworzy zaś wirtualne karty sieciowe, do których może zostać przyłączony sterownik protokołu. Gdy sterownik protokołu wysyła pakiet do sterownika pośredniego ten przesyła go do sterownika miniportu. W przeciwnym kierunku sterownik miniportu przesyła dane otrzymane z karty sieciowej do sterownika pośredniego, który następnie przesyła je do sterownika protokołu.

Sterowniki pośrednie używane są do:

- Translacji pakietów pomiędzy różnymi rodzajami mediów transmisyjnych. Przykładowo sterownik pośredni pomiędzy warstwą transportową Ethernet a sterownikiem miniportu ATM, przetwarza pakiety Ethernet na pakiety ATM i odwrotnie.
- Filtrowania pakietów. Przykładem takiego wykorzystania jest harmonogram pakietów. Moduł ten sprawdza priorytet pakietu wysłanego w dół stosu przez sterownik protokołu, lub w górę przez sterownik miniportu i ustala kolejność przesyłania pakietów ze względu na wartość priorytetu.
- Balansowania transmisją pakietów między większą ilością kart sieciowych. Dla sterownika protokołu udostępnia on pojedynczy wirtualny adapter, jednak ruch sieciowy kieruje na kilka kart sieciowych zwiększając prędkość przesyłania danych.

Sterownik protokołu to najwyżej położony w stosie NDIS sterownik i jest zarazem najniżej usytuowanym komponentem z zaimplementowanym protokołem sieciowym. Przykładem takiego sterownika jest sterownik TCP/IP lub IPX/SPX. Sterownik protokołu alokuje dla pakietu odpowiedni obszar w pamięci, kopiuje dane z aplikacji do przygotowanego pakietu i poprzez wywołanie funkcji NDIS przesyła go do karty sieciowej. Tworzy również interfejs dla przychodzących z karty sieciowej danych i przesyła je do aplikacji.

6. Podsumowanie

Zaprezentowana w artykule metoda oraz opracowany na jej podstawie algorytm pozwalają na wyznaczenie możliwych, bezkolizyjnych ścieżek przesyłania: komunikatów diagnostycznych informujących o stanie niezawodnościowym systemu, testów oraz przetwarzanych danych. Algorytm bazuje na strukturze diagnostycznej systemu i umożliwia wybranie ścieżki, która

nie będzie uwzględniała innych elementów aktualnie realizujących transmisję danych oraz tych, które znajdują się w stanie niezdatności. Przyjęte założenie można rozszerzyć o elementy realizujące w danym momencie proces przetwarzania danych, ale wymaga to przyjęcia dodatkowych, nie uwzględnionych w aktualnej wersji algorytmu, założeń.

Wykorzystanie omówionego w artykule sterownika pakietowego daje możliwość budowy sieci bazującej na różnych typach interfejsów sieciowych. Mogą to być zarówno połączenia kablowe (karty Ethernet, standard USB), jak też bezprzewodowe (Wi-Fi, Bluetooth).

Literatura

- [1] CHEN M.-S., SHIN K.G., *Depth-first search approach for faulttolerant routing in hypercube multicomputers*, IEEE Transactions Parallel and Distributed Systems, vol. 1, no. 2, str. 152-159, 1990.
- [2] CHIU G. M., WU S.P.: *A Fault-Tolerant Routing Strategy in Hypercube Multicomputers*, IEEE Transactions on Computers, vol. 45, no. 2, str 143-155, 1996.
- [3] CHUDZIKIEWICZ J.: *Sieci komputerowe o strukturze logicznej typu hipersześcianu*, Instytut Automatyki i Robotyki, Wydziału Cybernetyki WAT, Warszawa, 2002.
- [4] CHUDZIKIEWICZ J.: *Metoda wyznaczania m-optimalnych struktur opiniowania diagnostycznego dla sieci komputerowych typu hipersześcianu*, V Krajowa Konferencja Diagnostyka Urządzeń i Systemów, Ustroń, 2003.
- [5] CHUDZIKIEWICZ J.: *Wyznaczanie m-diagnozowalnych struktur typu PMC w systemach o zwiększonej odporności na uszkodzenia*, X Konferencja Systemów Czasu Rzeczywistego, Ustroń, 2003.
- [6] CHUDZIKIEWICZ J., MURAWSKI K.: *Wyznaczanie bezkolizyjnych dróg przesyłania danych w sieci teleinformatycznej o strukturze typu hipersześcianu*, Polskiego Towarzystwa Diagnostyki Technicznej, Diagnostyka nr 3(39), 2006.
- [7] Dokumentacja: *Microsoft Windows 2003 Driver Development Kit*, Microsoft 2003.
- [8] GORDON J.M., STOUT Q.F.: *Hypercube message routing in the presence of faults*, Proc. Third Conf. on Hypercube Concurrent Computers and Applications, vol. 1, str. 318-327, 1988.
- [9] KORZAN B.: *Elementy teorii grafów i sieci. Metody i zastosowania*, Wydawnictwo Naukowo-Techniczne, Warszawa, 1978.
- [10] KULESZA R.: *Podstawy diagnostyki sieci logicznych i komputerowych*, Instytut Automatyki i Robotyki, Wydział Cybernetyki WAT, Wydanie II, Warszawa, 2000.
- [11] ONEY W.: *Programming the Microsoft Windows Driver Model*, wydanie drugie, Microsoft Press, 2003.

The method of determining a non collision data transfer path in system on overmining structure

ABSTRACT: In this paper the author presents the method and the algorithm for determining non collision data transfer path in n -dimensional hypercube computer networks. The hypercube structures have properties of auto-reconfiguration of network structure depending on failures or requiring conditions for auto-reconfiguration. n -dimensional hypercube computer networks belong to the class of fault tolerant computer networks and they are highly complex for $n > 3$. Moreover, for Windows® operating systems, the tools and mechanisms were presented that make implementation of this algorithm easier.

KEYWORDS: hypercube, auto-diagnosis, faulty tolerance systems

Recenzent: prof. dr hab. inż. Roman Kulesza

Praca wpłynęła do redakcji: 26.06.2007