System informatyczny dla wieloetapowego rozpoznawania obiektów

Computer system for multi-stage object recognition

Swietłana Lebiediewa[1]

Treść. Przedmiotem pracy jest przedstawienie informatycznego systemu wspomagającego proces wieloetapowego podejmowania decyzji w procesie wieloetapowego rozpoznawania obiektów.
Przedstawiono ideę rozpoznawania wieloetapowego jako metodę polegającą na dekompozycji problemu decyzyjnego. Przedstawiono charakterystykę drzewa decyzyjnego i ciągu uczącego. Zaprezentowano bazę wiedzy i operacyjną bazę danych do rozpoznawania wieloetapowego. Omówiono współpracę algorytmów rozpoznawania z bazą danych.

Słowa kluczowe: baza danych, baza wiedzy, rozpoznawanie wieloetapowe.

Abstract. The subject of this paper is to present the specificity of computer system for systems of multistage decision making in the process multistage pattern recognition. The paper presents the idea of multi-stage recognition as a method involving the decomposition of the decision problem. The characteristics of the decision tree and the learning sequence are presented. Knowledge base and operational database to a multi-step recognition are presented.
Cooperation recognition algorithms to the database are discussed.

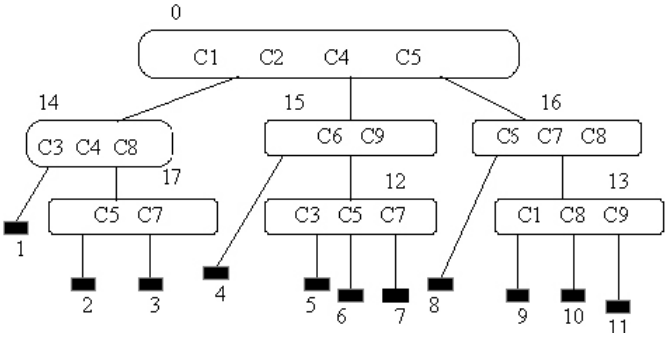
Keywords: database, multi-stage recognition, knowledge base.

1. Wstęp

W procesie podejmowania złożonych decyzji znaczne miejsce znajdują metody wykorzystujące podejścia polegające na dekompozycji problemu decyzyjnego. Dzięki takim dekompozycjom zazwyczaj uzyskujemy możliwość przetwarzania większych problemów decyzyjnych. Jednym ze stosowanych podejść jest podejmowanie wieloetapowych decyzji z wykorzystaniem drzewa decyzyjnego jako „szkieletu" prowadzenia zdekomponowanego procesu decyzyjnego. Przykładem takiego postępowania jest rozpoznawanie wieloetapowe, w którym zastępujemy jednorazowe rozpoznawanie sekwencją tzw. rozpoznawań lokalnych, których przeprowadzenie jest zgodne z zadaną konstrukcją drzewa decyzyjnego, tj. wzdłuż jednej z możliwych ścieżek korzeń-węzeł terminalny. Takie podejście pozwala uzyskać miarodajne rozpoznanie badanego obiektu, zazwyczaj z efektywniejszym niż w przypadku rozpoznawania jednorazowego, wykorzystaniem danych pomiarowych oraz pozostałych zasobów odpowiedniego systemu komputerowego. Metodyka rozpoznawania wieloetapowego została wprowadzona w pracy [1,3].

2. Rozpoznawanie wielostopniowe

Rozpoznawanie wielostopniowe (RW) obiektów [1, 2, 3] polega na tym, że w poszczególnych etapach rozpoznawania podejmuje się decyzję, do którego podzbioru wszystkich możliwych klas należy klasa rozpoznawanego obiektu oraz o tym, jakie cechy ze zbioru wszystkich możliwych cech mają być mierzone w następnym etapie rozpoznawania. Tak więc rozpoznawanie wielostopniowe polega na *dekompozycji* problemu decyzyjnego, czyli zastępowaniu jednorazowego rozpoznawania sekwencją tzw. rozpoznawań lokalnych, przeprowadzanych w poszczególnych węzłach zgodnie z zadaną konstrukcją **drzewa decyzyjnego** (DD), ryc.1.1. Żeby zaklasyfikować rozpoznawany obiekt do jednej z klas, należy przejść ścieżką w DD, startując od korzenia. W każdym napotykanym węźle wewnętrznym (**decyzyjnym**) należy podjąć decyzję o dalszej drodze w grafie, aż osiągnięty zostanie węzeł terminalny. Klasa z nim związana reprezentuje końcowy wynik rozpoznawania wielostopniowego [4].



Ryc. 1. 1. Drzewo decyzyjne

1. Wrocławska Wyższa Szkoła Informatyki Stosowanej „Horyzont", Wydział Informatyki ul. Wejherowska 28, 54-239 Wrocław, swietlana@lebiediewa.com

Algorytmy rozpoznawania wielostopniowego (ARW) z uczeniem korzystają z ciągu uczącego (CU), ryc 1.2. CU jest ciągiem par postaci (x_k, k), gdzie x_k jest wektorem wartości cech obiektu, k zaś – numerem klasy [2,3], symbol * w CU oznacza, że cecha jest nieistotna dla klasy i przy obliczaniu „odległości" nie jest brana pod uwagę. W dalszych rozważaniach będziemy rozpatrywać współpracę z CU *Algorytmu najbliższego sąsiada* (ANS). Algorytm ten polega na tym, że rozpoznawany obiekt klasyfikowany jest do tej samej klasy, do której należy element CU, którego „odległość" od rozpoznawanego obiektu jest najmniejsza („odległość" jest pewną normą w przestrzeni cech).

C1	C2	C3	C4	C5	C6	C7	C8	C9	KLASA
16	3,44	3	6,55	40,0	48	116	*	14,9	7
15	3,45	2	8,45	40,0	*	*	120	*	1
14	2,4	1	8,33	38,1	*	110	*	*	2
18	3,0	*	7,5	35,0	40	*	*	15,0	4
15	2,7	*	6,0	20,0	35	110	112	*	8

Ryc. 1.2. Fragment ciągu uczącego.

3. System informatyczny dla rozpoznawania wielostopniowego

System informatyczny dla rozpoznawania wielostopniowego zawiera elementy bazy wiedzy (BW), algorytmy rozpoznawania i operacyjną bazę danych (BD). Operacyjną BD przedstawiono w paragrafach 3 i 4, algorytmy rozpoznawania – w paragrafach 3 i 5.

Elementy bazy wiedzy to:

- Zbiór poprawnie sklasyfikowanych obiektów (zbiór faktów doświadczalnie zebranych i zweryfikowanych) reprezentowany przez elementy CU - wiedza pochodząca od eksperta;
- Drzewo decyzyjne (klasyfikator oparty na DD) pozwalające zdekomponować proces rozpoznawania na etapy - wiedza pochodząca od eksperta;
- Algorytmu rozpoznawania - wiedza pochodząca od eksperta.

CU ma postać tablicy prostokątnej, w ostatniej kolumnie, w której znajdują się numery klas obiektów, w pozostałych – wartości cech poprawnie sklasyfikowanych obiektów, ryc. 1.2.

Drzewo decyzyjne może być przedstawione jako spójny, niezorientowany, acykliczny graf skończonego stopnia (drzewa), z każdym węzłem którego jest związana pewna informacja, wykorzystywana przez algorytmy rozpoznawania. Informacja ta może być przedstawiona w postaci tablic, ryc.2.1 [5,6]

Tablica a) zawiera informacje o liczbie cech wykorzystywanych w węźle i liczbie bezpośrednich następników węzła. Tablica b) przyporządkowuje każdemu z węzłów nieterminalnych numery cech wykorzystywanych w tym węźle. Tablica c) przyporządkowuje węzłom nieterminalnym numery klas osiągalnych z tych węzłów. Tablica d) zawiera informację o numerach następników każdego węzła nieterminalnego wraz z informacją, czy następnik jest węzłem terminalnym czy nie.

a)

NR WĘZŁA	LICZBA CECH	LICZBA NASTĘPNIKÓW
0	4	2
14	3	2
11	2	2
15	2	2
12	3	3

b)

CECHA	WĘZEŁ
C1	0
C2	0
C3	14
C4	0
C5	0
C6	12

c)

WĘZEŁ	KLASA
17	2
17	3
14	1
14	2
14	3
12	5
12	6
15	4
15	5

d)

WĘZEŁ	NASTĘPNIK	WĘZEŁ TERMINALNY
0	14	0
0	15	0
14	1	1
14	17	0
17	2	1
17	3	1
15	4	1
15	12	0

Ryc. 2.1. Tablice opisujące strukturę DD.

Opisu drzewa decyzyjnego dostarcza ekspert. Ekspert określa również algorytmy rozpoznawania pozwalające na podstawie wartości zmierzonych cech zaliczyć obiekt do tej czy innej klasy.

4. Segment danych i algorytm rozpoznawania w węźle

Do podjęcia decyzji w węźle algorytm rozpoznawania potrzebuje następującej informacji: informacji o fragmencie CU i cechach obiektu wykorzystywanych w tym węźle, informacji o numerach klas osiągalnych z tego węzła, informacji o tym, czy wynik rozpoznawania w węźle (numer klasy) jest następnikiem danego węzła, a jeżeli nie, informacji, z którego węzła-następnika danego węzła klasa ta jest osiągalna. Informacja potrzebna do podjęcia decyzji w jednym węźle drzewa decyzyjnego znajduje się w jednym zbiorze danych. Zbiór taki nazywa się **segmentem** a identyfikator segmentu indeksuje się numerem i węzła decyzyjnego, z którym segment ten jest związany. Na każdym etapie rozpoznawania wieloetapowego algorytmy rozpoznawania współpracują z jednym segmentem danych, pobierając informację zawartą w segmencie i po przetworzeniu tej informacji, wpisując wyniki przetwarzania (podjęte decyzje) do segmentu. Tak więc segment danych stanowi autonomiczną jednostkę bazy danych do rozpoznawania wieloetapowego, a dla rozpoznawania jednoetapowego cała baza danych redukuje się do jednego segmentu. Niech symbol * oznacza numer węzła. Segment zawiera następującą informację [5]: Informację o cechach wykorzystywanych w węźle* (CECHY*), informację o następnikach węzła* (NASTĘPNIKI*), informację o klasach osiągalnych z węzła* (KLASY_OSIĄGALNE*), informację o fragmencie CU używanym w węźle* (FRAGSEQ*), informację o wartościach cech mierzonych w węźle* (TAB*), a także informację o decyzjach podjętych w tym węźle (DECYZJE*). Decyzją może być numer klasy lub numer węzła nieterminalnego, do którego trze-

ba przekazać sterowanie. W celu ilustracji przedstawimy bazę danych zredukowaną do jednego segmentu – relacyjnej BD zredukowanej do segmentu 14 na ryc.3.1.

NASTĘPNIKI_14

NR_ WĘZŁA	NR_WĘZŁA NASTĘPNIKA	T
14	1	TAK
14	17	NIE

CECHY_14

NR_CECHY	NR_WĘZŁA
C3	14
C4	0
C8	14

KLASY_OSIĄGALNE_14

NR KLASY	NR DECYZJI	T
1	1	TAK
2	17	NIE
3	17	NIE

FRGSEQ_14

C3	C4	C8	NR KLASY
3	6,58	128	1
4	6,31	145	3
1	6,12	137	3
2	6,75	186	2
5	6,43	129	1
3	6,13	135	2
2			1
	6,02	112	

DECYZJE 14 (fragment)

O–ID	NR_WĘZŁA	T
001	1	TAK
005	17	NIE
007	17	NE

TAB14 (fragment)

O-ID	C3	C4	C8
001	2	6,24	125
005	1	6,86	131
007	1	6,35	145

Ryc.3.1 Model bazy danych zredukowanej do segmentu SEG14

ALGORYTM ROZPOZNAWANIA W WĘŹLE* (symbol * oznacza numer węzła)

KROK 1. Wczytaj z BD identyfikator i wektor cech rozpoznawanego obiektu.

KROK 2. Wczytaj fragment CU FRAGSEQ* wykorzystywany w danym węźle.

KROK 3. (obliczanie wartości funkcji rozpoznawania wektora cech rozpoznawanego obiektu **g**): dla wszystkich elementów relacji FRAGSEQ*: pobierz element CU, oblicz normę euklidesową z rozpoznawanym obiektem, zapamiętaj normę euklidesową i numer klasy obiektu, której ona odpowiada.

KROK 4. Zapisz identyfikator obiektu i rezultat rozpoznawania w WĘŹLE* do relacji DECYZJE* w BD. **STOP**.

5. Model konceptualny i modele zewnętrzne bazy danych dla RW

Najprostszym rozwiązaniem jest przyjęcie modelu konceptualnego będącego sumą segmentów danych. Gdyby wszystkie zapytania do BD były związane z AR - wędrówką rozpoznawanego obiektu po DD - potraktowanie modelu konceptualnego jako sumy segmentów byłoby nie tylko naturalne ale i optymalne. W każdym segmencie łatwo zrealizować odpowiedzi na zapytania:
Podaj wektor cech wykorzystywany w węźle *;
Jakie klasy są osiągalne w węźle *;
Jakie decyzje podjęto w węźle * o obiekcie NR-Obiektu;
Jakie następniki ma węzeł *;
Do jakiego węzła należy przekazać sterowanie, jeżeli rezultatem decyzji jest klasa *i*;
Jaki węzeł jest poprzednikiem węzła *;
Ile następników ma węzeł *;
Podaj wszystkie cechy obiektu NR-Obiektu i rezultat rozpoznawania w węźle *.
Natomiast, jeżeli do BD REC będą kierowane zapytania nie związane bezpośrednio z AR, np. zapytania dotyczące struktury DD, przyjęcie modelu konceptualnego jako sumy segmentów byłoby kłopotliwe. Rzeczywiście, realizacja zapytań typu:
Podaj wektor cech DD;
Jaki jest poprzednik klasy *i;*
Podaj węzły na ścieżce (*węzeł-1, węzeł-2*);
Jakie obiekty należą do klasy *i*?
Wymagałoby całkowitego przeglądu BD. Dlatego przy projektowaniu modelu konceptualnego brano pod uwagę zarówno realizację zapytań związanych z rozpoznawaniem jak i realizację zapytań dotyczących struktury DD.
Tak więc w modelu konceptualnym bazy danych REC występują relacje opisujące strukturę drzewa decyzyjnego (relacje WĘZŁY, KLASY, NASTĘPNIKI i KLASY OSIĄGALNE), relacja CECHY zawierająca informację o cechach wykorzystywanych w poszczególnych węzłach i informację o tym, w którym węźle cecha była mierzona po raz pierwszy, relacja SEQUENCE opisująca ciąg uczący, relacje typu TAB zawierające wartości cech rozpoznawanych obiektów, relacje typu DECYZJE zawierające rezultaty decyzji podejmowanych w poszczególnych węzłach, a także relacja SEGMENTY zawierająca informację o relacjach występujących w poszczególnych segmentach. Relacja WĘZŁY przyporządkowuje numerom węzłów nieterminalnych nazwy segmentów danych, a także zawiera taką informację o węzłach nieterminalnych, jak liczba cech wykorzystywanych w tym węźle, liczba bezpośrednich następników węzła i liczba jego poprzedników. Na ryc 4.1 przedstawiono fragment schematu bazy danych. Klucze relacji zaznaczono symbolem #.

RELATION WĘZŁY(NR_WĘZŁA#, SEG_ID, LICZBA CECH, BEZPOŚREDNIE-NASTĘPNIKI, POPRZEDNIKI, BEZP_POPRZEDNIK)

RELATION SEGMENTY (SEG_D#, NR_WĘZŁ, CU, NAZWA_TAB, CECHY, NASTĘPNIKI, DECYZJE, KLASY_ OSIĄGALNE)

RELATION KLASY (KLASA#, NR_WĘZŁA)

RELATION NASTĘPNIKI (NR_WĘZŁA#, NR_WĘ-

ZŁA_POPRZEDNIK)
RELATION KLASY OSIĄGALNE (KLASA#, NR_ WĘZŁA#)
RELATION CECHY (CECHA#, NR_WĘZŁA#, PIERWSZE WYSTĄPIENIE)
RELATION SEQUENCE (C1#,C2#,C3#,C4#,C5#,C6#, C7#,C8#,C9#, KLASA)
RELATION TAB 0 (OB_ID#,C1,C2, , C4, C5)
RELATION DECYZJE 0 (OB_ID#, NR_WĘZŁA, T)

Ryc. 4.1. Schemat bazy danych dla rozpoznawania wieloetapowego (fragment)

Relacja KLASY opisuje zależność pomiędzy numerem klasy a numerem węzła będącego bezpośrednim poprzednikiem tej klasy, dla każdej klasy wskazany jest jej poprzednik – Ryc.4.2a,. Relacja NASTĘPNIKI zawiera informacje o bezpośrednich następnikach węzłów nieterminalnych (i informację o bezpośrednich poprzednikach węzłów nieterminalnych będących wartościami atrybutu NR_WĘZŁA_NASTĘPNIK) – Ryc. 4.2b. Relacje WĘZŁY, KLASY i NASTĘPNIKI łącznie opisują strukturę drzewa decyzyjnego. Relacje KLASY, KLASY OSIĄGALNE i NASTĘPNIKI, pozwalają określić dowolną ścieżkę w drzewie decyzyjnym. W relacji CECHY zawarta jest informacja, w których węzłach wykorzystywane są poszczególne cechy i w którym węźle cecha jest mierzona po raz pierwszy.
Relacja typu DECYZJE zawiera rezultaty decyzji podejmowanych przez algorytmy rozpoznawania dla kolejno rozpoznawanych obiektów. Relacja DECYZJE* (* jest numerem węzła nieterminalnego) ma trzy atrybuty: OB_ ID, NR_WĘZŁA i T - identyfikator obiektu, numer węzła będącego rezultatem decyzji podjętej w węźle * oraz informacja, czy rezultatem decyzji jest węzeł terminalny (klasa) czy nie. Informacja o rozpoznawanych obiektach znajduje się w relacjach typu TAB*, fragmenty relacji TAB14 i relacji DECYZJE14 oraz fragment relacji CECHY zredukowanej do cech wykorzystywanych w węźle 14 przedstawiono na Ryc. 3.1. Informacja o CU znajduje się w relacji SEQUENCE.

a) Relacja KLASY

KLASA	NR _ WĘZŁA
1	14
2	17
3	17
4	15
5	12
6	12
7	12
8	16
9	13
10	13
11	13

b) Relacja NASTĘPNIKI

NR_WĘZŁA_ NASTĘPNIKA	NR_WĘZŁA
17	14
12	15
13	16
14	0
15	0
16	0

Ryc.4.2a, b. Model konceptualny. Relacje KLASY i NASTĘPNIKI

c) Relacja KLASY OSIĄGALNE

NR_KLASY	NR_WĘZŁA
2	17
3	17
1	14
2	14
3	14
4	15
5	15
6	15
7	15
5	12
6	12
7	12
8	16
9	16
10	16
11	16
9	13
10	13
11	13
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0

Ryc 4.2c. Model konceptualny. Relacja KLASY OSIĄGALNE

Oprócz modelu konceptualnego w bazie danych istnieją modele zewnętrzne – fragmenty BD, z którymi pracuje użytkownik - jeden lub kilka segmentów danych.

6. Współpraca algorytmów rozpoznawania wieloetapowego z bazą danych

Przed rozpoczęciem rozpoznawania trzeba dokonać otwarcia sesji, BD i wszystkich relacji modelu konceptualnego. Oznaczmy przez C_N – bieżący węzeł, przez C_SEG_ID – bieżący identyfikator segmentu, R_N niech oznacza numer węzła–korzenia DD a D_N – numer węzła będącego rezultatem rozpoznawania. Liczbę rozpoznawanych obiektów oznaczamy zmienną *W.* Przedstawimy ARW współpracujący z BD:

KROK 0.
OPEN S; **OPEN** DB; **OPEN** R; **READ** R_N. [Otwarcie sesji, BD i wszystkich relacji modelu konceptualnego. Odczytanie z BD numeru węzła–korzenia DD]. C_N:=RN; COUNTER:=0.

KROK 1.
CREATE SEGMENT; [Utworzenie segmetu (model zewnętrzny BD) dla aktualnego węzła]

KROK 2. **OPEN** R_SEG [otwarcie wszystkich relacji w segmencie].
KROK 3. **PROJECT** INFSEG* **ON** LICZBA_CECH **GIVING**C_LICZBA_CECH **SELECT** TAB* **WHERE** O_ID=C_O_ID **GIVING** C_TAB; **PROJECT** C_TAB **ON** O_ID **GIVING** C_ ID.
KROK 4. **CALL ALGORYTM ROZPOZNAWANIE-_W_WĘŹLE*.**
KROK 5. **SELECT** DECYZJE* **WHERE** O_ID= C_O_ID **GIVING** C_DEC; **PROJECT** C_DEC **ON** WĘZEŁ **GIVING** D_N.
KROK 6. **IF** D_N=”węzeł terminalny” **THEN GOTO** KROK 8;
KROK 7. C_N=D_N; **GOTO** KROK1.
KROK 8. COUNTER:=COUNTER+1; C_N=R_N; **IF** (COUNTER<w) **THEN GOTO** KROK 1 **ELSE CLOSE** S; **STOP.**

W kroku 0 następuje otwarcie sesji, bazy danych i wszystkich relacji modelu konceptualnego oraz nadanie wartości początkowych zmiennym. W kroku 1 zostaje utworzony segment danych dla aktualnego węzła: Zostają utworzone relacje INFSEG*, WĘZEŁ*, CECHY*, NASTĘPNIKI*, KLASY_OSIĄGALNE*, FRAGSEQ*. W kroku 2 zostają otwarte wszystkie utworzone relacje segmentu SEG* a także relacje TAB* i DECYZJE*. W kroku 3 zostaje pobrana informacja o liczbie cech wykorzystywanych w węźle *. W kroku 4 wywołuje się algorytm rozpoznawania w węźle. W kroku 5 następuje pobranie z relacji DECYZJE* wiersza, zawierającego identyfikator rozpoznawanego obiektu i numer węzła (podjętej decyzji), następnie zapamiętanie decyzji w zmiennej D_N. W następnych krokach sprawdza się, czy rezultat rozpoznawania jest węzłem terminalnym (co oznacza koniec rozpoznawania) i pobiera się następny element do rozpoznawania. Jeżeli rezultat rozpoznawania jest węzłem nieterminalnym, to ten węzeł (rezultat rozpoznawania) czyni się węzłem aktualnym i do niego zostaje przekazane sterowanie.

Operacje występujące w algorytmach rozpoznawania można podzielić na dwie grupy: instrukcje języka relacyjnej bazy danych i instrukcje realizujące funkcję rozpoznawania (np. obliczające “odległość” (normę euklidesową) rozpoznawanego obiektu od elementów CU i wybierając najmniejszą z tych “odległości”). Złożoność obliczeniową funkcji rozpoznawania dla ANS można oszacować w sposób następujący. Przyjmijmy, że długość wektora cech w węźle * równa się l. Do obliczenia metryki używa się jednej operacji odejmowania, jednej operacji mnożenia, (l-1) operacji dodawania i jednej operacji wyciągnięcia pierwiastka kwadratowego, razem 3*l operacji arytmetycznych. Następnie wykonuje się (m-1) operacji porównania *(m* – długość FRAGSEQ*). Ostatecznie do obliczenia funkcji rozpoznawania w węźle potrzeba 3*l*(m-1) operacji. Najbardziej czasochłonną operacją z pierwszej grupy jest operacja tworzenia modelu zewnętrznego (**CREATE** SEGMENT). Oznaczmy przez n długość (liczbę wierszy) najdłuższej relacji modelu konceptualnego. Niech * oznacza numer węzła. Relacje CECHY*, NASTĘPNIKI*, WĘZEŁ* i INFSEGMENT* powstają z odpowiednich relacji modelu konceptualnego za pomocą operacji **SELECT**. Złożoność obliczeniowa każdej z tych operacji jest $O(n)$. Relacja FRAGSEQ* powstaje z relacji SEQUENCE przez superpozycję operacji **SELECT** i **PROJECT**. Złożoność obliczeniową operacji **PROJECT** jest $O(m^2)$, gdzie m liczba wierszy relacji będącej argumentem operacji **PROJECT**. Niech s będzie długością najdłuższej ścieżki w drzewie. Wtedy algorytm rozpoznawania dla jednego obiektu wymaga s operacji **CREATE** SEGMENT, s–krotnego wywołania ALGORYTMU ROZPOZNAWANIA W WĘŹLE*, oraz wykonania kilkunastu relacyjnych operacji. Dla s=m pesymistyczna złożoność obliczeniowa jest więc $O(m^2+m)$, czyli – $O(m^2)$. ALGORYTM ROZPOZNAWANIA W WĘŹLE* pracuje na modelu zewnętrznym, co daje znaczną oszczędność czasu. Zamiast porównywać zmierzone cechy rozpoznawanego obiektu z każdym elementem CU, porównuje się je z elementami fragmentu CU FRAGSEG*, który jest krótszy od CU. Oznaczmy przez K liczbę węzłów terminalnych DD (liczbę klas), przez LE – liczbę elementów CU dla jednej klasy, a przez NK_i – liczbę klas osiągalnych z węzła nieterminalnego o numerze i. Współpraca algorytmu rozpoznawania z CU SEQUENCE występującym w modelu konceptualnym wymaga $K*LE$ porównań wektora cech rozpoznawanego obiektu z elementami CU, a współpraca z CU FRAGSEQ* występującym w modelu zewnętrznym wymaga NK_i*LE porównań. Dla jednego tylko węzła oszczędność wynosi $(K - NK_i)*LE$ porównań. Przykładowo, przy założeniu, że liczba elementów CU dla każdej klasy równa się 10, praca z całym CU dla drzewa z ryc.2.1 wymaga 11*10 = 110 porównań w każdym węźle, a praca z FRAGSEQ* wymaga 110 porównań w węźle 0, 30 porównań w węźle 14 i 20 porównań w węźle 17. Oszczędność dla każdego obiektu z klasy 1 na ścieżce (0,14) wynosi 110*2-(110+30)=80, a na ścieżce (0,14,17) – 110*3-(110+30+20)=170 porównań.

7. Uwagi końcowe

System ekspertowy jest systemem komputerowym używającym modeli wiedzy i procedur wnioskowania w celu rozwiązywania problemów o dużej skali trudności [7]. Podstawowa idea działania systemów ekspertowych polega na przeniesieniu wiedzy eksperta z określonej dziedziny do bazy wiedzy i posiadaniu posiadających informacji zaprojektowaniu maszyny wnioskującej [8]. Pełna realizacja takiego systemu pozwala umiejscowić go w klasie systemów ekspertowych przeznaczonych do wspomagania wypracowania zautomatyzowanej decyzji diagnostycznej [9]. Jednym z podstawowych problemów przy konstrukcji takiego systemu ekspertowego jest stworzenie i eksploatacja jego bazy wiedzy z faktami (CU, DD) i zaprojektowaniu maszyny wnioskującej realizującej algorytmy rozpoznawania.

Literatura (References)

[1] Z. Bubnicki, *Knowlege-Based Approach as a Genegalization of Pattern Recognition Problems and Methods. Systems Science,* 1993, Vol. 19, No 2, pp. 5-21.
[2] M. Flasiński, Wstęp do sztucznej inteligencji, PWN, Warszawa 2011.
[3] M. Kurzyński, Algorytmy rozpoznawania wieloetapowego oraz ich zastosowania medyczne i techniczne, Wyd. PWr. ,Wrocław 1987.
[4] J. Józefczyk, Rozpoznawanie i zastosowania biomedyczne, [w:] Problemy automatyki i informatyki, Wrocław: Wyd. Ossolineum 1998, 45-58.
[5] S. Lebiediewa, B. Węglorz, Problemowo zorientowane bazy danych w procesie wielostopniowego podejmowania decyzji, [w:] Współczesne problemy informatyki, Legnica 2011, 127-142.
[6] S. Lebiediewa, Metodologia projektowania problemowo zorientowanych baz danych do systemów wielostopniowego podejmowania decyzji, Wyd. PWr. 1998.
[7] J. Kisielnicki, H. Sroka, Systemy informatyczne biznesu, Warszawa 2005.
[8] L. Rutkowski, Metody i techniki sztucznej inteligencji, Warszawa, Wyd. PWN 2009.
[9] J. Józefczyk, Systemy z reprezentacją wiedzy. Systemy ekspertowe, [w:] Problemy automatyki i informatyki, Wrocław, Wyd. Ossolineum 1998, 71-82.

Efektywne programowanie w Matlabie. Odwracanie macierzy trójprzekątniowych metodą eliminacji Gaussa

Effective programming in Matlab.
Inverting tridiagonal matrices using Gaussian elimination

Paweł Keller[1], **Iwona Wróbel**[2]

Streszczenie: Celem cyklu artykułów *Efektywne programowanie w Matlabie* jest prezentacja sposobów pisania bardzo wydajnych algorytmów w języku Matlab, rozwiązujących wybrane problemy obliczeniowe. W niniejszym artykule przedstawiamy efektywną implementację metody eliminacji Gaussa zastosowanej do wyznaczania odwrotności macierzy trójprzekątniowych. Zaimplementowane zostały warianty eliminacji zarówno bez, jak i z wyborem elementów głównych. Wysoka efektywność stworzonych funkcji potwierdzona jest wykonanymi testami obliczeniowymi.

Słowa kluczowe: efektywne programowanie, Matlab, macierz odwrotna, macierz trójprzekątniowa, eliminacja Gaussa

Abstract: The series *Effective programming in Matlab* is meant to present very fast implementations of algorithms for solving various computational problems in the Matlab programming language. In this paper, we present a very efficient implementation of the Gaussian elimination algorithm applied to computing the inverse of a tridiagonal matrix. Two variants of the elimination, without and with pivoting, are considered. The high efficiency of the presented solutions is supported by computational examples.

Keywords: effective programming, Matlab, matrix inverse, tridiagonal matrix, Gaussian elimination

1. Introduction

We consider the problem of computing (in Matlab) the inverse of a nonsingular tridiagonal matrix $A \in \mathbb{R}^{n\times n}$, $A(i,j) = 0$ for $|j-i| > 1$. The problem is very interesting for two reasons. A tridiagonal matrix is a matrix of the simplest structure whose inverse is, in general, a full square matrix. In addition, in Matlab there are no build-in subroutines dedicated for computing inverses of matrices of such a type.

At this point we should note, that in many problems, where the inverse of a matrix appears, there is, in fact, no need to actually compute the inverse. Usually the problem can be solved by computing a solution of a corresponding system of linear equations or the corresponding matrix equation. Sometimes, however, the inverse itself needs to be computed.

Obviously, for computing the inverse of a tridiagonal matrix, one can try to use one of the subroutines for computing the inverse of a general square matrix. In Matlab, this can be done in several ways, e.g., `A^(-1)`, `inv(A)`, `A\eye(n)`, where n is the size of a matrix A (the `eye(n)` function creates the identity matrix of size n). The last one of the above ways is the fastest.

In the first experiment we have set $n = 20000$, generated a random tridiagonal matrix $A \in \mathbb{R}^{n\times n}$, and run the following Matlab code:

```
X = A \ eye(n);
```

where by X we denote (in the whole paper) the numerically computed inverse of the matrix A. The result appeared after two minutes (all the experiments presented in this paper were performed on the four core, 3.7GHz computer running 64-bit version of Matlab R2012b). The long computation time is no surprise, as Matlab solved the matrix equation

$$AX = I,$$

[1]Faculty of Computer Science, Wroclaw School of Information Technology, ul. Wejherowska 28, 54-239 Wroclaw, Poland; p.keller@horyzont.eu, p.keller@mini.pw.edu.pl

[2]Faculty of Mathematics and Information Science, Warsaw University of Technology, pl. Politechniki 1, 00-661 Warsaw, Poland; i.wrobel@mini.pw.edu.pl

using Gaussian elimination with pivoting, not being aware of a special form of the matrix A, and therefore used $O(n^3)$ operations to compute the inverse.

In the next section we shall introduce the Matlab data type called *sparse matrix*, which will help us to easily reduce the computation time of the algorithm based on the Matlab build-in functions to only $O(n^2)$ arithmetic operations.

In Section 3, we shall write our own function in the Matlab programming language. We shall use the simple Gaussian elimination algorithm and try to make our solution as efficient as possible.

In Section 4 we shall modify the fastest solution derived in the previous section, to use Gaussian elimination with pivoting, in order to make the function as accurate as the one based on the Matlab build-in operations, and, hopefully, significantly faster.

Some final efficiency tests and conclusions will be presented in Section 5.

2. Sparse matrices

In Linear Algebra, a *sparse matrix* is a matrix in which the number of nonzero elements is very small compared to the total number of elements. Sparse matrices are natively implemented in the Matlab system. To create a sparse matrix we usually use the `sparse` function (see [4] for more information) in one of several overloaded forms[1]. When a sparse matrix is created, Matlab stores only its nonzero elements (and locations of these elements in the matrix), in the columnwise order. When an operation involving a sparse matrix is performed, Matlab ignores the zero elements, unless it would change the result.

Using sparse matrices, we may easily write a short function that computes the inverse of a tridiagonal matrix in only $O(n^2)$ time. The function is presented in Listing 1. If we run this function to compute the inverse of a random tridiagonal matrix $A \in \mathbb{R}^{n\times n}$, where, as earlier, $n = 20000$, then the result will be obtained in only 6.5 seconds.

The function can be made even a little more efficient if we use more sophisticated way to create the sparse version of the input matrix. Using the

```
sparse(A)
```

command, we make Matlab search the whole matrix A for nonzero elements. We may speed up the process by pointing exactly which elements have to be stored in the sparse version of the matrix A. To this end, we have to tell the `sparse` function the row indices and column indices of all nonzero elements, and the elements themselves, in the columnwise order. The modified function is presented in Listing 2, and it computes the inverse of a tridiagonal 20000×20000 matrix in 6.1 seconds.

Listing 1. *The Matlab function that computes the inverse of a tridiagonal matrix using sparse matrices and "\" operator.*

```
function X = inv3(A)
%inv3    Tridiagonal matrix inverse.
%   inv3(A) is the inverse of the matrix A.
%   Sparse matrix and the left division "\" are used.

n = size(A,2);  % matrix size

% Making A sparse and computing the inverse...
X = sparse(A) \ eye(n);
```

Listing 2. *The Matlab function that computes the inverse of a tridiagonal matrix using explicitly created sparse matrix and the "\" operator. Please note that we have intentionally put the creation of the 3rd row of the matrix M at the beginning, in order to pre-allocate the memory space at the same time.*

```
function X = inv3s(A)
%inv3s    Tridiagonal matrix inverse.
%   inv3s(A) is the inverse of the matrix A.
%   "Hand made" made sparse matrix is used.

n = size(A,2);  % matrix size

% Preparing column-wise ordered values...
M(3,:) = [ diag(A,-1); 0 ];
M(1,:) = [ 0; diag(A,1) ];
M(2,:) = diag(A,0);
% Column indices: [1:n; 1:n; 1:n]
q = repmat( 1:n, 3, 1 );
% Row indices: [-1;0;1] * ones(1,n) + q
r = repmat([-1;0;1], 1, n) + q;
% Making the sparse version of the input matrix...
H = sparse(r(2:end-1), q(2:end-1), M(2:end-1), n, n);

% Computing the inverse...
X = H \ eye(n);
```

One can say that the time saved (only 6%) is not worth complicating the function code. However, we have done that for one more reason. In practice, a tridiagonal matrix is never stored as a full square one. Usually the superdiagonal, diagonal and subdiagonal elements are given as three separate vectors or one $3 \times n$ (or $n \times 3$) matrix. Following the way we create the sparse matrix (see Listing 2) we choose to remember the elements of the input matrix A in the matrix $M \in \mathbb{R}^{3\times n}$ in such a way that k'th column contains all nonzero elements of the k'th column of A. More precisely,

$$M(i,k) = A(k-2+i,k), \quad i \in \{1,2,3\}$$

(where we assume that $A(0,1) = A(n,n+1) = 0$), or in a little more verbose way,

$$\begin{aligned} M(1,k) &= A(k-1,k), \\ M(2,k) &= A(k,k), \\ M(3,k) &= A(k+1,k). \end{aligned} \tag{2.1}$$

Using (2.1) we may write our final implementation of the function that computes the inverse of a tridiagonal matrix using sparse matrices and Matlab left division operator. The function is given in Listing 3.

[1] It means that the function `sparse` works differently, depending on the type and the number of input arguments.

Listing 3. *The Matlab function that computes the inverse of a tridiagonal matrix using explicitly created sparse matrix and the "\" operator. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3v(M)
%inv3v    Tridiagonal matrix inverse.
%   inv3v(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Sparse matrix and the left division are used.

n = size(M,2);  % matrix size

% Making the sparse version of the input matrix...
% Row indices: [-1;0;1] * ones(1,n) + [1:n; 1:n; 1:n]
% Column indices: [1:n; 1:n; 1:n]
q = repmat( 1:n, 3, 1 );
r = repmat([-1;0;1], 1, n) + q;
H = sparse( r(2:end-1), q(2:end-1), M(2:end-1), n, n );

% Computing the inverse...
X = H \ eye(n);
```

Storing the input matrix in a memory efficient way neither increases nor decreases the performance time of the corresponding function. The 20000×20000 matrix is still inverted in 6.1 seconds. For a better clarity of the listed Matlab functions, in all the listings, we skip the argument checking part.

3. Simple Gaussian elimination

In this section, we shall write the function for inverting tridiagonal matrices, completely from the scratch, using Gaussian elimination without pivoting. If we are succeeded in creating significantly more efficient solution than the one given in Listing 3, then in the next section we shall extend the function to use Gaussian elimination with pivoting, so that our algorithm never fails.

The simple (without pivoting) Gaussian elimination algorithm is a well known and a very simple way to solve the system of linear equations

$$Ax = b \tag{3.1}$$

(see [1, §1.3], [2, §4.2] for more details). The algorithm becomes even simpler, if A is a tridiagonal matrix. The following Matlab code, under some additional assumptions on the matrix A (that LU factorisation of A exists, see [1, §1.3]), solves the system (3.1):

```
n = size(A,1);

% Elimination phase...
for k = 1:n-1
  A(k+1,k+1) = A(k+1,k+1) - A(k+1,k)/A(k,k) * A(k,k+1);
  b(k+1) = b(k+1) - A(k+1,k)/A(k,k) * b(k);
end

% Back substitution phase...
x(n) = b(n) / A(n,n);
for k = n-1:-1:1
  x(k) = ( b(k) - A(k,k+1)*X(k+1) ) / A(k,k);
end
```

All we need to do now is to adapt the above algorithm to solve the matrix equation

$$AX = I \tag{3.2}$$

instead of the system (3.1). This means, that instead of manipulating on the elements of the vector b, we have to perform analogous operations on the whole rows of the identity matrix I, or, in fact, only on those elements of each row, that actually change.

The function that solves the equation (3.2) in the case of a tridiagonal matrix A, i.e. the function that inverts a tridiagonal matrix, using simple Gaussian elimination, is given in Listing 4. In order to save memory space, as in the function `inv3v` in Listing 3, the elements of a matrix being inverted are stored in the $3 \times n$ array, according to formulae (2.1).

Listing 4. *The Matlab function that computes the inverse of a tridiagonal matrix using simple Gaussian elimination algorithm. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vnp(M)
%inv3vnp    Tridiagonal matrix inverse.
%   inv3vnp(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Simple Gaussian elimination is used.

n = size(M,2);  % matrix size

E = eye(n);  % identity matrix

% Elimination phase...
for k = 1:n-1
  M(2,k+1) = M(2,k+1) - M(3,k)/M(2,k) * M(1,k+1);
  E(k+1,1:k) = E(k+1,1:k) - M(3,k)/M(2,k) * E(k,1:k);
end

% Back substitution phase...
X(n,:) = E(n,:) / M(2,n);
for k = n-1:-1:1
  X(k,:) = ( E(k,:) - M(1,k+1)*X(k+1,:) ) / M(2,k);
end
```

Sadly, the last function requires 16 seconds to invert a random 20000×20000 matrix, and is much slower than the function from Listing 3, based on the build-in Matlab operations. We may observe that the instruction

```
E(k+1,1:k) = E(k+1,1:k) - M(3,k)/M(2,k) * E(k,1:k);
```

(of the elimination phase) can be simplified to

```
E(k+1,1:k) = -M(3,k)/M(2,k) * E(k,1:k);
```

with no influence on the result. This is because, before the instruction is performed, the vector $E(1:k, k+1)$ contains only zeroes. With this modification, the computation time drops to 12.7 seconds, but this is still much too slow.

In order to considerably speed up our new function, we have to recall a very important fact related to the way Matlab uses to store matrices in the computer memory: the matrices are stored columnwise

(see [3, 8.5]). Considering the architecture of the modern CPUs, all computations are much more effective if they are performed on a continuous block of data. The function in Listing 4 works on rows of matrices E and X, and the elements in rows are scattered in the memory. Thus, our first optimisation step is to modify the function to perform vector operations only on the columns of the two matrices, instead of on the rows.

If the matrix X is the inverse of a matrix A, then it has to satisfy two equations, (3.2) and

$$XA = I. \tag{3.3}$$

Using the Gaussian elimination to solve the equation (3.3) we shall obtain the algorithm in which only the column operations are performed. The algorithm can be easily derived form the one in Listing 4, if we observe that the equation (3.3) is equivalent to

$$A^T X^T = (XA)^T = I^T = I.$$

From relations (2.1), we can easily see that transposing the matrix A, i.e. exchanging the elements $A(k+1,k)$ with $A(k,k+1)$, is equivalent to exchanging the elements $M(3,k)$ and $M(1,k+1)$ in the corresponding matrix M. Thus the function inverting a tridiagonal matrix by solving the equation (3.3) may look like the one in Listing 5.

Listing 5. *The Matlab function that computes the inverse of a tridiagonal matrix using simple Gaussian elimination and column operations. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vnpc3(M)
%inv3vnpc3    Tridiagonal matrix inverse.
%   inv3vnpc3(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Simple Gaussian elimination with column
%   operations is used.

n = size(M,2);  % matrix size

E = eye(n);  % identity matrix

% Elimination phase...
for k = 1:n-1
  M(2,k+1) = M(2,k+1) - M(1,k+1)/M(2,k) * M(3,k);
  E(1:k,k+1) = -M(1,k+1)/M(2,k) * E(1:k,k);
end

% Back substitution phase...
X(:,n) = E(:,n) / M(2,n);
for k = n-1:-1:1
  X(:,k) = ( E(:,k) - M(3,k)*X(:,k+1) ) / M(2,k);
end
```

The efficiency gain is quite noticeable. The new function, which performs column operations, requires only 2.9 seconds to compute the inverse of a tridiagonal matrix $A \in \mathbb{R}^{20000\times 20000}$, i.e. is more than twice as fast as the best solution based on Matlab build-in subroutines. We shall try, however, to make the function even more effective. The next optimisation will not be strictly related to the Matlab language.

Observe that there is no need to use extra memory space for the identity matrix E (in Listing 5). In addition, creation of such a matrix also consumes time. Thus, we shall use only one matrix variable to store the identity matrix and to store the final result. This can be easily done, as shown in Listing 6. Now, the computation time drops to 2 seconds, in the case of a random 20000×20000 tridiagonal matrix.

Listing 6. *The Matlab, memory optimised, function that computes the inverse of a tridiagonal matrix using simple Gaussian elimination and column operations. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vnpc2(M)
%inv3vnpc2    Tridiagonal matrix inverse.
%   inv3vnpc2(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Simple Gaussian elimination with column
%   operations is used. Memory optimised version.

n = size(M,2);  % matrix size

X = eye(n);  % initialising main variable

% Elimination phase...
for k = 1:n-1
  M(2,k+1) = M(2,k+1) - M(1,k+1)/M(2,k) * M(3,k);
  X(1:k,k+1) = -M(1,k+1)/M(2,k) * X(1:k,k);
end

% Back substitution phase...
X(:,n) = X(:,n) / M(2,n);
for k = n-1:-1:1
  X(:,k) = ( X(:,k) - M(3,k)*X(:,k+1) ) / M(2,k);
end
```

The last optimisation step may be a little surprising. We shall increase the number of arithmetic operations to decrease the computation time. During the elimination phase (see Listing 6), we modify the elements $X(i,k+1)$ only for $1 \le i \le k$. This is because the elements $X(i,k)$ for $i > k$ are all equal zero, and there is no need to consider them. Thanks to this, the cost of the elimination phase is only $n^2/2 + O(n)$ arithmetic operations.

On the other hand, in Matlab, addressing a part of a matrix is, in most cases, done quite inefficiently. We shall illustrate this by the following example. Assume n is the number of rows of the matrix X. Let us consider the two instructions

```
Y = X(1:n,k);
```

and

```
Y = X(:,k);
```

In both cases the result is exactly the same. However, the latter is about 2 times faster. In the second instruction Matlab addresses the k'th column of X directly, and thus it is done very fast. In the first instruction of the above, Matlab makes no optimisations. This means that first, a vector $[1,2,\ldots,n]$ of indices is created, and only then a new vector containing the elements $X(i,k)$ for $i \in [1,2,\ldots,n]$ is formed, which is finally assigned to the variable Y.

For the above reason, it may be more time efficient to address the whole columns of X during the elimination phase of our algorithm, even though it increases the number of arithmetic operations of this phase to $n^2 + O(n)$. The final version of the function that inverts a tridiagonal matrix using Gaussian elimination without pivoting is presented in Listing 7. Indeed, to compute the inverse of a random 20000×20000 tridiagonal matrix we now need only 1.6 seconds.

Note (see Listing 7), that we have to additionally restore the diagonal element (equal 1) which is overwritten in the case the whole columns of X are assigned the new values.

Listing 7. *The final version of the Matlab function computing the inverse of a tridiagonal matrix using simple Gaussian elimination and column operations. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vnpc(M)
%inv3vnpc2    Tridiagonal matrix inverse.
%   inv3vnpc(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Simple Gaussian elimination with column
%   operations is used. Fully optimised version.

n = size(M,2);  % matrix size

X = eye(n);  % initialising main variable

% Elimination phase...
for k = 1:n-1
  M(2,k+1) = M(2,k+1) - M(1,k+1)/M(2,k) * M(3,k);
  X(:,k+1) = -M(1,k+1)/M(2,k) * X(:,k);
  X(k+1,k+1) = 1;
end

% Back substitution phase...
X(:,n) = X(:,n) / M(2,n);
for k = n-1:-1:1
  X(:,k) = ( X(:,k) - M(3,k)*X(:,k+1) ) / M(2,k);
end
```

4. Gaussian elimination with pivoting

The simple Gaussian elimination not always can be performed. The algorithm will fail if any of the elements $A(k,k)$, i.e. $M(2,k)$ in our implementation $(1 \leq k < n)$, become zero. Also, if this element is small, but different from zero, large roundoff errors may appear, making the computed matrix inverse very inaccurate (see, e.g., [1, §1.3] for more details).

There are classes of matrices for which we can use the simple Gaussian elimination with no risk of failure or instability, e.g., for the class of diagonally dominant matrices. A tridiagonal matrix $A \in \mathbb{R}^{n\times n}$ is called *diagonally dominant*, if

$$|A(k,k)| > |A(k-1,k)| + |A(k+1,k)| \quad (1 \leq k \leq n)$$

or

$$|A(k,k)| > |A(k,k-1)| + |A(k,k+1)| \quad (1 \leq k \leq n).$$

In such a case the function `inv3vnpc` from Listing 7 is very fast and very accurate.

In the case of a general tridiagonal matrix, some pivoting technique has to be applied to the Gaussian elimination algorithm, in order to obtain a stable method for inverting tridiagonal matrices. We shall not discuss the pivoting scheme in details. It is clearly explained, e.g., in [2, §4.3]. As, in our algorithm, we solve the matrix equation (3.3), during the elimination phase, we shall look for the largest (in modulus) element in consecutive rows, and exchange the corresponding columns, if necessary. The function that inverts a tridiagonal matrix using Gaussian elimination with pivoting is given in Listing 8. Note that the function requires that $n \geq 2$.

Listing 8. *The Matlab function that computes the inverse of a tridiagonal matrix using Gaussian elimination algorithm with pivoting and column operations. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vppc2(M)
%inv3vnpc2    Tridiagonal matrix inverse.
%   inv3vnpc(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Gaussian elimination with pivoting
%   and column operations are used.

n = size(M,2);  % matrix size

M(4,n) = 0;  % additional diagonal ("sub-sub-diagonal")

X = eye(n);  % initialising main variable

% Elimination phase...
for k = 1:n-1
  % Pivoting...
  if abs(M(1,k+1)) > abs(M(1,k))
     T = M(2:4,k);
     M(2:4,k) = M(1:3,k+1);
     M(1:3,k+1) = T;
     X(:,[k,k+1]) = X(:,[k+1,k]);
  end
  % Elimination...
  M([2,3],k+1) = M([2,3],k+1)...
                 - M(1,k+1)/M(2,k) * M([3,4],k);
  X(:,k+1) = X(:,k+1) - M(1,k+1)/M(2,k) * X(:,k);
end

% Back substitution phase...
X(:,n) = X(:,n) / M(2,n);
X(:,n-1) = ( X(:,n-1) - M(3,n-1)*X(:,n) ) / M(2,n-1);
for k = n-2:-1:1
  X(:,k) = ( X(:,k) - X(:,[k+1,k+2])*M([3,4],k) )...
           / M(2,k);
end
```

The pivoting increased the computation time, in the case of a random 20000×20000 matrix, to 3.7 seconds. This is mostly because we wrote the function in Listing 8 according to the (Matlab) book. As it was shown at the end of Section 3, sometimes less vectorised code may be more efficient, if we can save some background matrix creation operations.

In the function from Listing 8, a $2 \times n$ matrix is created $2n-3$ times (when we refer to $X(:,[k+1,k])$ in the pivoting phase, and to $X(:,[k+1,k+2])$ in the back substitution phase). Therefore, in the next, and our last function (see Listing 9), we have tried to avoid all possible to avoid background matrix creation operations. To this end, we have changed several

vectorised instructions to the equivalent set of scalar ones. The resulting function has significantly longer code, but is over 35% faster than the previous one. Our test matrix is inverted in 2.4 seconds.

Listing 9. *The final version of the Matlab function that computes the inverse of a tridiagonal matrix using Gaussian elimination algorithm with pivoting and column operations. The elements of a matrix being inverted are stored in the $3 \times n$ array.*

```
function X = inv3vppc(M)
%inv3vnpc2    Tridiagonal matrix inverse.
%   inv3vnpc(M) is the inverse of the tridiagonal
%   matrix A whose nonzero elements are given
%   in the 3-by-n matrix M as follows:
%   M(i,k) = A(k-2+i,k), i = 1,2,3.
%   Gaussian elimination with pivoting and column
%   operations are used. Matlab-optimised version.

n = size(M,2);  % matrix size

M(4,n) = 0;  % additional diagonal ("sub-sub-diagonal")

X = eye(n);  % initialising main variable

% Elimination phase...
for k = 1:n-1
  % Pivoting...
  if abs(M(1,k+1)) > abs(M(2,k))
     T1 = M(2,k);
     T2 = M(3,k);
     M(2,k) = M(1,k+1);
     M(3,k) = M(2,k+1);
     M(4,k) = M(3,k+1);
     M(1,k+1) = T1;
     M(2,k+1) = T2;
     M(3,k+1) = 0;
     T = X(:,k);
     X(:,k) = X(:,k+1);
     X(:,k+1) = T;
  end
  % Elimination...
  M(2,k+1) = M(2,k+1) - M(1,k+1)/M(2,k) * M(3,k);
  M(3,k+1) = M(3,k+1) - M(1,k+1)/M(2,k) * M(4,k);
  X(:,k+1) = X(:,k+1) - M(1,k+1)/M(2,k) * X(:,k);
end

% Back substitution phase...
X(:,n) = X(:,n) / M(2,n);
X(:,n-1) = ( X(:,n-1) - M(3,n-1)*X(:,n) ) / M(2,n-1);
for k = n-2:-1:1
  X(:,k) = ( X(:,k) - X(:,k+1)*M(3,k)...
                    - X(:,k+2)*M(4,k) ) / M(2,k);
end
```

5. Final experiments and conclusions

In this section, we will compare the functions `inv3v` (Listing 3) and `inv3vppc` (Listing 9) with respect to the accuracy. We shall also make a more thorough time comparison test of the two above functions and the simple `A\eye(n)` Matlab instruction.

As the function `inv3v` solves the equation (3.2) to compute the inverse, while the function `inv3vppc` solves the (3.3) one, to make the accuracy test fair, for each generated random matrix $A \in \mathbb{R}^{100\times100}$ we also invert its transposition. We measure the error

$$\mathcal{E}(A) = \frac{\max\left\{||AX - I||_2, ||XA - I||_2\right\}}{\mathrm{cond}_2(A)}, \qquad (5.1)$$

where $||\cdot||_2$ is the second matrix norm, and $\mathrm{cond}_2(\cdot)$ is the corresponding condition number. We define the error in the above way, because a good inverse should satisfy both, (3.2) and (3.3), equations.

In Table 1 we present the results obtained for 1500000 random tridiagonal matrices of size 100. As we can see, the functions are equally accurate, which is no surprise, as both are based on the same method, Gaussian elimination with pivoting.

Table 1. Comparison of the $\mathcal{E}(A)$ error (5.1) of the functions `inv3v` and `inv3vppc`, for 1500000 random tridiagonal matrices $A \in \mathbb{R}^{100\times100}$.

function	average error	maximum error
`inv3v`	$1.7 \cdot 10^{-16}$	$2.7 \cdot 10^{-13}$
`inv3vppc`	$1.7 \cdot 10^{-16}$	$1.5 \cdot 10^{-13}$

In the final test, we compare the efficiency of the functions `inv3v`, `inv3vppc`, and the Matlab `A\eye(n)` instruction (see Section 1), for random tridiagonal matrices of different sizes, from 32 to 16384. The results are presented in Figure 1.

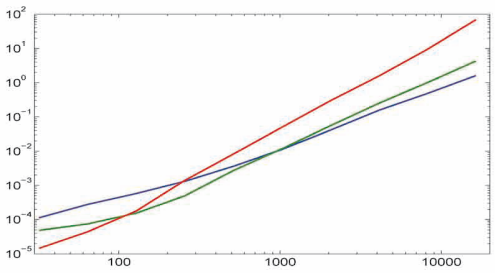


Figure 1. Dependency of the computation time of the Matlab `A\I` instruction (red), the function `inv3v` (green) and the function `inv3vppc` (blue), on the matrix size (displayed in the logarithmic scale).

Matlab is an interpretive language, and therefore a user function always has a huge efficiency disadvantage compared to the build-in one. Thus, for small matrices, the solutions based on the Matlab "\" operator are faster. However, thanks to the optimizations we have made, in the case of large matrices, our function wins, being almost 3 times faster, if a matrix size is greater than 20000. Were the efforts made worth the result? We leave the answer to the reader.

References

[1] G. Dahlquist and Å. Björck, *Numerical Methods in Scientific Computing: Volume 1*, SIAM, 2008.

[2] D. Kincaid and W. Cheney, *Numerical Analysis*, Brooks/Cole, 1993.

[3] *Getting Started with MATLAB*, Matlab documentation, MathWorks, 2007.

[4] `http://www.mathworks.com/help/matlab/`.

Analiza działania protkokołu STP dla portów zagregowanych

The analysis of STP protocol for aggregated interfaces

Radosław Wróbel[1], Tomasz Ortyl[1],
Adam Sobkowiak[1], Waldemar Kokot[1]

Treść. Celem przedstawionych prac było sprawdzenie, czy protokoły agregujące wiele portów fizycznych w jeden logiczny (LACP, PAgP) oraz protokół STP (RSTP) na skonsolidowanych kanałach, pracują zgodnie z dokumentacją IEEE i Cisco. Badania przeprowadzono, agregując kilka portów (dokładnie: 1, 2 i 4) i sprawdzając wynik działania protokołu drzewa rozpinającego. W pracy przedstawiono nie tylko wyniki działania protokołów (komendy show), ale także sposób konfiguracji. Badania wykonano w laboratorium Cisco Wrocławskiej Wyższej Szkoły Informatyki Stosowanej, wykorzystując przełączniki rodziny Cisco Catalyst 2960.

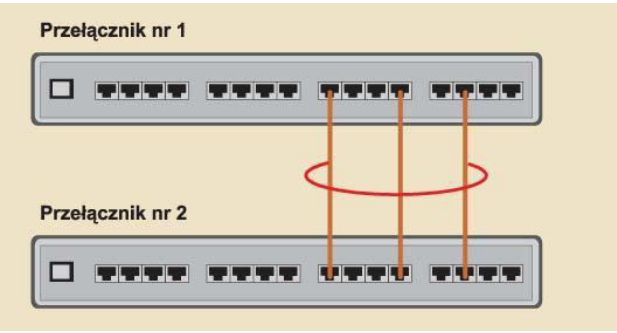
Słowa kluczowe: STP, agregacja portów, przełącznik.

Abstract. The aim of showed researchers was getting confirmation about proper working protocols, aggregated few physical interfaces into one, and checking of STP (RSTP) protocol works on consolidated channels. Researchers was making by aggregation few ports (exactly: 1, 2 and 4) and checked results of Spanning – Tree Protocols works. In paper showed not only results of working protocols ("show" commands) but also proper manner of configuration for channel consolidation. The researchers was made in Cisco laboratory of Wroclaw School of Information Technology , on new switches Catalyst 2960

Key words: STP, Port aggregation, switch.

1. Wstęp

Agregacja portów z wykorzystaniem *Port Channel* jest popularną metodą zwiększania niezawodności i przepustowości połączeń fizycznych, wykonywaną w systemach autonomicznych, wykorzystujących przełączniki zarządzalne. Mogłoby się wydawać, że podłączenie trunkowe dwóch lub większej ilości przełączników przy pomocy kilku, jednocześnie załączonych kabli automatycznie wywoła procesy protokołu STP (*ang. Spanning-Tree Protocol*) i tak jest w rzeczywistości. STP wyłącza nadmiarowe ścieżki (niekiedy w sposób, nie wprost widoczny dla administratora), aby uniknąć zapętleń, a co za tym idzie powielenia zapytań i odpowiedzi, aż do stanu, który uniemożliwi funkcjonowanie sieci.



Rys. 1. Porty skonsolidowane przy pomocy Port Channel [1]

Agregacja, o której mowa (nazywana niekiedy konsolidacją), polega na fizycznym podłączeniu kilku portów przełącznika z kilkoma portami innego przełącznika (lub innego urządzenia obsługującego tą technologię). Maksymalna liczba jednocześnie wykorzystywanych połączeń fizycznych, które mogą uczestniczyć w agregacji wynosi 8 [1, 2]. Łączne pasmo logicznego połączenia jest równe sumie pasm poszczególnych połączeń fizycznych, uczestniczących w procesie, a grupa interfejsów fizycznych, traktowana jest, jako pojedynczy, logiczny interfejs.
W artykule przedstawiono analizę działania protokołu drzewa rozpinającego (STP) na grupie portów, zagregowanych przy pomocy *Port Channel*. Przedstawiono wyniki działania protokołu STP dla różnej ilości skonsolidowanych portów.

2. Protokoły: STP, PAgP i LACP

STP

Spanning Tree Protocol (STP) jest to nieroutowalny protokół warstwy 2 modelu ISO/OSI opisany w standardzie IEEE 802.1D, działa w obrębie domeny rozgłoszeniowej segmentu sieci LAN. Swoje działanie opiera na protokole

1. Wrocławska Wyższa Szkoła Informatyki Stosowanej „Horyzont", Wydział Informatyki, ul. Wejherowska 28, 54-239 Wrocław, rwrobel@horyzont.eu

drzewa rozpinającego *(ang. Spanning-Tree Protocol)*. Każdy przełącznik przechowuje topologię sieci w obrębie swojej domeny rozgłoszeniowej, zbudowaną na postawie protokołu STP.

Zadania realizowane przez STP:

-zapewnienie środowiska wolnego od pętli w przypadku, gdy topologia fizyczna jak i logiczna segmentu sieci LAN posiada połączenia nadmiarowe,

-zmiana trasy ramek warstwy 2 na połączenie nadmiarowe w przypadku utraty połączenia podstawowego,

-zapewnia, że w fizycznym i logicznym segmencie sieci LAN pomiędzy dwoma urządzeniami warstwy 2 jest tylko jedno aktywne połączenie.

Ogólna zasada działania STP.

Po uruchomieniu protokołu STP wybierany jest przełącznik, który stanie się korzeniem drzewa rozpinającego *(ang. rootem)* topologii. Aby to umożliwić, na wszystkich portach przełączników wysyłane są ramki konfiguracyjne protokołu STP, nazwane Bridge Protocol Data Units (BPDU). Po wybraniu przełącznika, który będzie rootem, na pozostałych następuje wybór drogi do roota, który dokonywany jest na postawie kosztu danej trasy. Port, który jest na trasie o najniższym łącznym koszcie do roota, staje się root portem i przechodzi w tryb przekazywania *(ang. forwarding)*. Port lub porty, które prowadzą **od** roota i są na trasie o wyższych kosztach, zostają wybrane na porty desygnowane *(ang. designated ports)* i są również w trybie forwarding. Pozostałe porty są w stanie blokowania *(ang. blocked)*. Przykładowo koszt trasy 100Mbps to 19, a trasy 10Gbps to 2 [3].

Po zaniku lub zwiększeniu kosztu trasy na root porcie, STP na podstawie swojej bazy topologii zmienia port o dotychczas gorszej trasie na root port, stan portu zmienia się z blocked na forwarding (designated port na przełączniku bliżej roota jest cały czas w trybie forwarding) i alternatywna trasa staje się aktywna. W przypadku takiej zmiany topologii wysyłane są ramki BPDU Topology Change Notification, które informują o tym, aby topologię przebudować. Do utrzymania aktualności topologii wykorzystywane są pakiety BPDU, wysyłane co dwie sekundy przez roota i przekazywane w głąb drzewa [3].

W tabeli 1 poniżej stany oraz domyślne czasy przejścia do następnego stanu, przez jakie przechodzi port przełącznika korzystający z STP [3].

Tab. 1. Stany przez jakie przechodzi port przełącznika pracujący w domyślnej konfiguracji

Stan portu	Odbieranie BPDU	Wysyłanie BPDU	Czy znany MAC adres urządzenia podłączonego do portu?	Czy port uczestniczy w transmisji danych?
1) Disabled	Nie	Nie	Nie	Nie
2) Blocking (20 sekund)	Tak	Nie	Nie	Nie
3) Listening (15 sekund)	Tak	Tak	Nie	Nie
4) Learning (15 sekund)	Tak	Tak	Tak	Nie
5) Forwarding	Tak	Tak	Tak	Tak

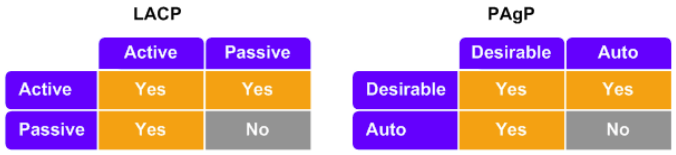
LACP i PAgP

Ogólne właściwości.

LACP (IEEE 802.3ad) i PAgP (protokół własności Cisco) są to nieroutowalne protokoły pozwalające na agregację do ośmiu fizycznych połączeń pomiędzy dwoma urządzeniami w jedno połączenie logiczne znane jako Port channel bądź etherchannel. Realizują one rozkładanie ruchu na zagregowanym logicznym połączeniu na wszystkie fizyczne połączenia. Takie logiczne połączenie może być skonfigurowanie jako trunk lub jako access. Protokoły te wymagają, aby interfejsy fizyczne uczestniczące w linku logicznym miały takie same ustawienia m.in. co do szybkości i duplexu. Połączenie logiczne działa tak długo jak długo aktywne jest przynajmniej jedno połączenie fizyczne [6].

Rys. 2. Przykład zestawionego połączenia

Poniżej znajduje się zobrazowanie w jakiej sytuacji może dojść do zestawienia zagregowanego kanału.



Rys. 3. Zasada poprawnego zestawienia zagregowanego połączenia [4]

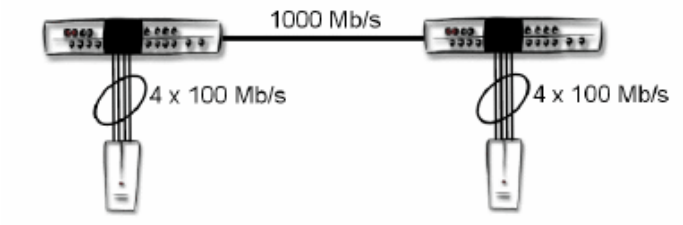
Zasadnicze różnice.

PAgP – *(ang. Port Aggregation Protocol)*

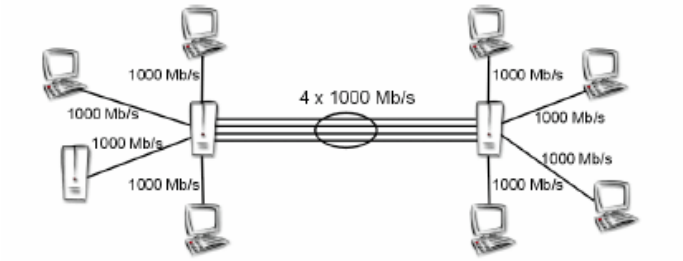
- Protokół własności Cisco.
- Pakiety PAgP wysyłane są zawsze do dobrze znanego adresu multicastowego urządzeń Cisco tj.01-00-0C-CC-CC-CC [7].
- STP komunikuje się pierwszym aktywnym połączeniem fizycznym [7].

LACP – (*ang. Link Aggregation Control Protocol)*

- Umożliwia dodanie do zagregowanego połączenia dodatkowych 8 fizycznych połączeń jako zapasowe.
- W porównaniu do PAgP możliwy do skonfigurowania również na hostach np.



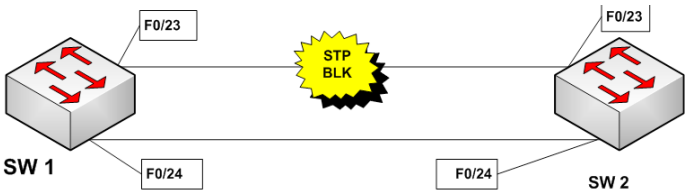
Rys. 4. Połączenie przełącznik – host [5].



Rys. 5. Połączenie host – host [5].

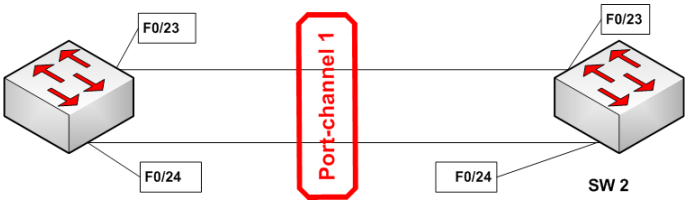
3. Przebieg eksperymentu

Do przeprowadzenia testów wykorzystano dwa 24 portowe przełączniki Ethernet firmy Cisco model WS-C2960. W zależności od etapu testu wykorzystywano od jednego do ośmiu jednoczesnych połączeń pomiędzy przełącznikami. Na rys. 6 przedstawiono wynik zadziałania protokołu STP, w wyniku którego port nadmiarowy (redundantny) został przełączony w stan blokowania.



Rys. 6. Połączenie przełączników dwoma równoległymi łączami trunkowymi i wynik działania protokołu STP

Jak można zauważyć, na powyższym rysunku, jedno z połączeń fizycznych zostało przez protokół STP przełączone w stan BLK (*BLOCKED*) i stanowi połączenia alternatywne na wypadek awarii głównego połączenia. W tym wypadku, aby ominąć (skądinąd pożądane) działanie protokołu STP, wykonano sieć testową, agregując obydwa porty. Rys. 7 przedstawia ideowe działanie sieci po tej operacji.



Rys. 7. Połączenie przełączników dwoma równoległymi połączeniami zagregowanymi w jedno logiczne połączenie.

W tabeli 1 zaprezentowano konfigurację, charakterystyczną dla Port Channel, która umożliwia implementację technologii.

Tab. 2. Detale konfiguracyjne przy agregacji portów

SW1	SW2
hostname SW1 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface FastEthernet0/23 channel-group 1 mode desirable ! interface FastEthernet0/24 channel-group 1 mode desirable	hostname SW2 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface FastEthernet0/23 channel-group 1 mode auto ! interface FastEthernet0/24 channel-group 1 mode auto

Jak można zauważyć (tabela 2), interfejsy FastEthernet0/23 i FastEthernet0/24 przynależą do channel-grupy 1, będącej reprezentantem zagregowanego interfejsu *Port-Channel1*. Jako protokół sygnalizacyjny został ustalony protokół PAgP. Wyświetlając stan interfejsu Port-channel1, można potwierdzić przynależność dwóch wskazanych interfejsów do zagregowanego linku:

```
SW2# sh int port-channel1
Port-channel1 is up, line protocol is up (connected)
(Hardware is EtherChannel, address is 0022.bef3.e898 (bia
0022.bef3.e898)
 ,MTU 1500 bytes, BW 200000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Full-duplex, 100Mb/s, link type is auto, media type is unk-
nown
 input flow-control is off, output flow-control is unsuppor-
ted
 Members in this channel: Fa0/23 Fa0/24
ARP type: ARPA, ARP Timeout 04:00:00
```

Natomiast status protokołu STP potwierdza równoległą pracę obu połączeń:

```
SW1#sh spanning-tree
VLAN0001
Spanning tree enabled protocol rstp
Root ID    Priority    32769
Address     001a.6dda.af80
This bridge is the root
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
(Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
Address     001a.6dda.af80
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time 300
Interface        Role Sts Cost      Prio.Nbr Type
-------------------------------- ---------- ---------- --------
Po1              Desg FWD 12        128.72   P2p
```

W następnym kroku w grupie zagregowanych portów znalazły się dwa *Port Channele*, zawierające po dwa por-

ty (w tym wypadku: Fe0/1, 0/2, 0/23 i 0/24). W tabeli 3 zaprezentowano konfigurację, charakterystyczną dla Port Channeli (po dwa porty), która umożliwia implementację technologii.

Tab. 3. Detale konfiguracyjne przy agregacji portów (po 2 porty w kanale)

SW1	SW2
hostname SW1 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface Port-channel2 ! interface FastEthernet0/1 channel-group 2 mode active ! interface FastEthernet0/2 channel-group 2 mode active ! interface FastEthernet0/23 channel-group 1 mode desirable ! interface FastEthernet0/24 channel-group 1 mode desirable	hostname SW2 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface Port-channel2 ! interface FastEthernet0/1 channel-group 2 mode passive ! interface FastEthernet0/2 channel-group 2 mode passive ! interface FastEthernet0/23 channel-group 1 mode auto ! interface FastEthernet0/24 channel-group 1 mode auto

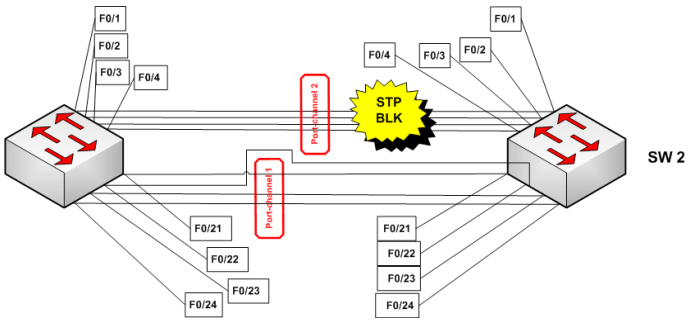
Jak widać, interfejsy FastEthernet0/23 i FastEthernet0/24 przynależą do channel-grupy 1 będącej reprezentantem zagregowanego interfejsu Port-Channel1. Jako protokół sygnalizacyjny został ustalony protokół PAgP. Interfejsy FastEthernet 0/1 i FastEthernet 0/2 przynależą do channel-grupy 2 będącej reprezentantem zagregowanego interfejsu Port-Channel2. Jako protokół sygnalizacyjny został ustalony protokół LACP. Wyświetlając stan interfejsu Port-channel1 można potwierdzić przynależność dwóch wskazanych interfejsów do zagregowanego linku:

```
SW2# sh int port-channel1
(Port-channel1 is up, line protocol is up (connected)
(Hardware is EtherChannel, address is 0022.bef3.e898
(bia 0022.bef3.e898)
 ,MTU 1500 bytes, BW 200000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Full-duplex, 100Mb/s, link type is auto, media type
 is unknown
 input flow-control is off, output flow-control is
unsupported
 Members in this channel: Fa0/23 Fa0/24
ARP type: ARPA, ARP Timeout 04:00:00
```

Z punktu widzenia protokołu STP, agregowane połączenie jest traktowane jako jeden logiczny link. Dodanie drugiego agregowanego połączenia powoduje, że protokół STP potraktuje jedno z nich jako połączenie alternatywne i zablokuje je. Dowodem na to jest status protokołu STP na przełącznikach, natomiast zagregowany interfejs Port-channel 2 traktowany jest jako alternatywny (dla SW1):

```
SW1#sh spanning-tree
VLAN0001
Spanning tree enabled protocol rstp
Root ID    Priority    32769
Address     001a.6dda.af80
This bridge is the root
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
(Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
Address     001a.6dda.af80
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time 300
Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------
-------------
Po1              Desg FWD 12        128.72   P2p
Po2              Desg FWD 12        128.80   P2p
```

Aby potwierdzić działanie protokołu przy większej ilości portów, wykonano i zaprojektowano sieć (rys. 8) dla czterech portów w kanale. Rysunek przedstawia zdarzenie, polegające na blokowaniu pojedynczego kanału przez protokół RSTP.



Rys. 8. Połączenie przełączników dwoma równoległymi połączeniami (po 4 interfejsy), zagregowanymi w jedno logiczne połączenie

W następnym kroku w grupie zagregowanych portów znalazły się dwa interfejsy typu *Port Channel*, zawierające po dwa porty (w tym wypadku: Fe0/1:0/4 i 0/21:0/24). W tabeli 4 zaprezentowano konfigurację, charakterystyczną dla interfejsów Port Channel (po cztery porty), która umożliwia implementację technologii.

Jak widać na poniższym przykładzie konfiguracji, interfejsy FastEthernet0/21, FastEthernet 0/22, FastEthernet0/23 i FastEthernet0/24 przynależą do channel-grupy 1 będącej reprezentantem zagregowanego interfejsu Port-Channel1. Jako protokół sygnalizacyjny został ustalony protokół PAgP. Interfejsy FastEthernet 0/1, FastEthernet 0/2, FastEthernet0/3 i FastEthernet0/4 przynależą do channel-grupy 2 będącej reprezentantem zagregowanego interfejsu Port-Channel2. Jako protokół sygnalizacyjny został ustalony protokół LACP.

Tab. 4. Detale konfiguracyjne przy agregacji portów (po 2 porty w kanale)

SW1	SW2
hostname SW1 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface Port-channel2 ! interface FastEthernet0/1 channel-group 2 mode active ! interface FastEthernet0/2 channel-group 2 mode active ! interface FastEthernet0/3 channel-group 2 mode active ! interface FastEthernet0/4 channel-group 2 mode active ! interface FastEthernet0/21 channel-group 1 mode desirable ! interface FastEthernet0/22 channel-group 1 mode desirable ! interface FastEthernet0/23 channel-group 1 mode desirable ! interface FastEthernet0/24 channel-group 1 mode desirable	hostname SW2 ! spanning-tree mode rapid-pvst spanning-tree extend system-id ! interface Port-channel1 ! interface Port-channel2 ! interface FastEthernet0/1 channel-group 2 mode passive ! interface FastEthernet0/2 channel-group 2 mode passive ! interface FastEthernet0/3 channel-group 2 mode passive ! interface FastEthernet0/4 channel-group 2 mode passive ! interface FastEthernet0/21 channel-group 1 mode auto ! interface FastEthernet0/22 channel-group 1 mode auto ! interface FastEthernet0/23 channel-group 1 mode auto ! interface FastEthernet0/24 channel-group 1 mode auto

Wyświetlając stan interfejsu Port-channel1, można potwierdzić przynależność dwóch wskazanych interfejsów do zagregowanego linku:

```
SW2# sh int port-channel1
(Port-channel1 is up, line protocol is up (connected)
(Hardware is EtherChannel, address is 0022.bef3.e898
(bia 0022.bef3.e898)
 ,MTU 1500 bytes, BW 200000 Kbit, DLY 100 usec,
reliability 255/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Full-duplex, 100Mb/s, link type is auto, media type
is unknown
 input flow-control is off, output flow-control is
unsupported
Members in this channel: Fa0/21 Fa0/22 Fa0/23 Fa0/24
ARP type: ARPA, ARP Timeout 04:00:00
```

Z punktu widzenia protokołu STP agregowane połączenie nadal jest traktowane jako jeden logiczny link. Dodanie drugiego agregowanego połączenia powoduje, że protokół STP potraktuje jedno z nich jako połączenie alternatywne i zablokuje je. Dowodem na to jest status protokołu STP na przełącznikach, który zagregowany interfejs Port-channel 2 traktuje jako alternatywny:

```
SW1#sh spanning-tree
VLAN0001
Spanning tree enabled protocol rstp
Root ID    Priority    32769
Address     001a.6dda.af80
This bridge is the root
Hello Time   2 sec  Max Age 20 sec Forward Delay 15 sec
(Bridge ID  Priority    32769  (priority 32768 sys-id-ext 1)
Address     001a.6dda.af80
Hello Time   2 sec  Max Age 20 sec  Forward Delay 15 sec
Aging Time 300
Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- ------------
Po1              Desg FWD 12        128.72   P2p
Po2              Desg FWD 12        128.80   P2p
```

4. Podsumowanie

Celem badań było potwierdzenie działania protokołów agregujących w kontekście ich współdziałania z protokołem STP. Badanie potwierdziło założenia, że z punktu widzenia protokołu STP (RSTP), skonsolidowane połączenia są zgodnie z dokumentacją i traktowane jako pojedyncze połączenie logiczne. Całość przeprowadzonych eksperymentów potwierdza zgodność implementacji protokołów w platformie użytej do przeprowadzenia badań tj. przełączniki Cisco model 2960.

Literatura (References)

[1] Strona domowa Computerworld. *http://www.computerworld.pl/artykuly/319724/ Agregacja.portow.miedzy.przelacznikami.html.*
[2] Cisco: Configuring Port Channels. *http://www.cisco.com/c/en/us/td/docs/ switches/datacenter/nexus5000/sw/configuration/guide/cli/CLIConfigurationGuide/EtherChannel.html.*
[3] CCNP: Building Cisco Multilayer Switched Networks Study Guide (642-811) by Terry Jack (Nov 5, 2003).
[4] EtherChannel considerations *http://packetlife.net/blog/2010/jan/18/etherchannel-considerations/.*
[5] Link Aggregation according to IEEE Standard 802.3ad *http://cs.uccs.edu/~scold/doc/linkageaggregation.pdf*
[6] IEEE 802.3ad Link Aggregation.
[7] EtherChannels & Load-Balance *http://ericleahy.com/index.php/etherchannels/.*

SYSTEM EKSPERTOWY A DOBÓR KADR W PRZEDSIĘBIORSTWIE
Część B: Realizacja Systemu Ekspertowego

The Expert System vs the Staff Selection. Part B: The creation of an expert system

Karolina Plawgo[2], Marian Czerwiński[1]

Treść. Praca jest kontynuacją publikacji [15]. Zaproponowano nowy system ekspertowy. Zastosowano reguły i odpowiadające im współczynniki stanowią oryginalne podejście do problemu doboru kadr. Wykorzystano pewne rozwiązania systemu ekspertowego szkieletowego, opisanego w [13]. System wyróżnia się stosunkowo niskim kosztem realizacji oraz dużą efektywnością w stosunku do innych technik selekcji kadry. Zaproponowane podejście minimalizuje wpływ czynników subiektywnych na ostateczną ocenę kandydata. System może być stosowany do doboru kadry na różne stanowiska w większości organizacji.

Słowa kluczowe: System ekspertowy, dobór kadr, system wnioskujący, bazy wiedzy, wnioskowanie progresywne, wnioskowanie regresywne.

Abstarct. The work is a continuation of the publication [15]. In this paper new expert system is proposed. The rules were applied as well as the corresponding coefficients. This is an original approach to the problem of personnel selection. Although some solutions from skeleton expert system described in [13] were used. The system is distinguished by a relatively low cost of implementation and high efficiency compared to other personnel selection techniques. The proposed approach minimizes the influence of subjective factors on the final evaluation of the candidate. The system can be used for selection of personnel for the various positions in most organizations.

Keywords: expert system, personnel selection problem, requesting system, knowledge database, progressive inference, regressive inference

1. Wprowadzenie

W Części A omówiono podział kadry ze względu na kwalifikacje oraz inne cechy. Na tej podstawie w części B publikacji zaproponowano metodę doboru kadry, wykorzystującą system ekspertowy szkieletowy, opisany przez Antoniego Niederlińskiego [13], którą wyróżnia stosunkowo niski koszt realizacji oraz wysoka użyteczność w stosunku do innych technik selekcji kadry dla potrzeb przedsiębiorstwa. W zaprojektowanym systemie ekspertowym zastosowano reguły i odpowienie współczynniki pewności. Podejście to minimalizuje wpływ czynników subiektywnych, gdyż ostateczna ocena kandydata nie zależy w pełni od osoby odpowiedzialnej za końcowy etap rekrutacji.

Trzeba również zaznaczyć, iż prawdopodobieństwo zatrudnienia właściwego pracownika nie stosując żadnej z technik selekcji jest niskie, nie przekracza dwudziestu procent, natomiast z wykorzystaniem poprawnie skonstruowanego systemu ekspertowego jest czterokrotnie wyższe.

Wykorzystanie zaproponowanego systemu ekspertowego może przyczynić się do zatrudniania osób, które z większym prawdopodobieństwem będą mogły sprostać czekającym na nie zadaniom, prowadząc w rezultacie do wzrostu satysfakcji z wykonywanej pracy.

System ekspertowy jest stosowany do tych zagadnień, które są słabo sformalizowane, tzn. w których trudno jest przypisać teorie oparte na matematyce, lub inaczej, do których trudno jest zastosować ścisłe algorytmy działania. Przykładami tutaj mogą być: zarządzanie, medycyna, geologia, rolnictwo, prawo, astronautyka, robotyka, chemia, architektura.

2. Metody konstrukcji systemów ekspertowych

Systemy ekspertowe są programami komputerowymi przeznaczonymi do rozwiązywania specjalistycznych problemów wymagających profesjonalnej ekspertyzy. Są one w zasadzie skonstruowane w ten sposób, że wiedza dotycząca danej dziedziny jest odseparowana od reszty systemu. W programach nie będących systemami eks-

1. Wrocławska Wyższa Szkoła Informatyki Stosowanej „Horyzont", Wydział Informatyki, ul. Wejherowska 28, 54-239 Wrocław, mczerwinski@horyzont.eu
2. Wrocławska Wyższa Szkoła Informatyki Stosowanej „Horyzont", Wydział Informatyki, ul. Wejherowska 28, 54-239 Wrocław, karolina.plawgo@horyzont.eu

pertowymi (np. proceduralnych), wiedza dziedzinowa i wiedza o sposobie rozwiązania problemu są splątane i powiązane, najczęściej w bardzo nieczytelny sposób. Taka budowa programów komplikuje sytuację, iż ekspert dziedzinowy czytający kod takiego programu najczęściej nie potrafi stwierdzić, co ów program zakłada o rozwiązywanym problemie. Oznacza to również, że ów ekspert dziedzinowy musi korzystać z pomocy specjalisty-informatyka dla odczytania wiedzy dziedzinowej zawartej w programie jak również dla wprowadzania tejże wiedzy do programu. Sytuacja taka stwarza poważne trudności przy tworzeniu dużych, specjalizowanych programów, wymagających stosowania wiedzy dziedzinowej szeregu ekspertów, nie do uniknięcia staje się bowiem przynajmniej częściowe przekształcenie się specjalisty-informatyka w niezbyt doskonałego eksperta dziedzinowego lub eksperta dziedzinowego w niezbyt doskonałego informatyka.

Podstawową zaletą systemu ekspertowego jest oddzielenie bazy wiedzy (a więc części programu opisującej wiedzę dziedzinową związaną z rozwiązywanym problemem) i systemu wnioskującego (a więc części programu rozwiązującej problem).

Z takiej własności systemów ekspertowych wynika szereg ważnych korzyści – możliwość posługiwania się wiedzą dziedzinową w postaci jawnej (tzn. w postaci pliku tekstowego), co w znaczący sposób ułatwia odczytanie i zrozumienie wiedzy dziedzinowej, zastosowanej w programie. Zaleta ta jest bardzo istotna, gdyż dzięki temu czytelność wiedzy dziedzinowej, z której korzystają złożone programy, stworzone do podejmowania odpowiedzialnych decyzji, ma duże znaczenie zarówno dla użytkowników tych programów (którzy mogą zobaczyć z jakich zasobów wiedzy korzysta program) jak i dla ekspertów dziedzinowych, będących źródłem tej wiedzy (którzy mogą w prosty sposób wiedzę stosowaną przez program zweryfikować).

Kolejną własnością omawianych systemów jest możliwość ciągłego rozwijania baz wiedzy - na każdym etapie tworzenia dodaje się nowe elementy bazy wiedzy do tych, które już istnieją.

Niebłahą zaletą jest również możliwość tworzenia systemów ekspertowych skorupowych, tzn. uniwersalnych systemów ekspertowych pozbawionych wiedzy dziedzinowej. Są to tzw. rules based systems. Mamy tu już do czynienia z czymś, co można określić mianem sztuki komputerowej, dzięki której system zostaje w maksymalnym stopniu dopasowany do indywidualnych cech decydenta. Systemy takie mogą być w stosunkowo łatwy sposób wykorzystane przez różnych użytkowników poprzez wymianę bazy wiedzy. W ten sposób staje się możliwe tworzenie specjalistycznych programów przez ekspertów dziedzinowych nie będących specjalistami-informatykami. Systemy ekspertowe skorupowe są więc środowiskiem programowym umożliwiającym ekspertom dziedzinowym tworzenie – na drodze nieprogramowej – pełnowartościowych programów rozwiązujących ich problemy, bez konieczności zagłębiania się w informatyczne subtelności [13].

Wymóg opisu jakiejkolwiek działalności za pomocą zrozumiałych, jasnych i przejrzystych reguł, jest czymś dyscyplinującym ową działalność i stanowiącym warunek doskonalenia tej działalności. Patrząc w ten sposób na systemy ekspertowe wydaje się oczywiste, iż są to narzędzia umożliwiające czy wręcz wymuszające (dzięki zawartym w nich regułom) doskonalenie czynności, którym są dedykowane. System do doboru kadry w tym ujęciu formalizuje proces oceny kandydata czyniąc go bardziej przejrzystym.

Jak już wcześniej zostało powiedziane, systemy ekspertowe są programami komputerowymi przeznaczonymi do rozwiązywania specjalistycznych problemów (najczęściej o charakterze niealgorytmicznym) z pewnej dziedziny wymagających profesjonalnej ekspertyzy, wykonujące złożone zadania o dużych wymaganiach intelektualnych i robiące to tak dobrze jak człowiek, będący ekspertem w tej dziedzinie. Określenie system ekspertowy może być zastosowane do dowolnego programu komputerowego, który na podstawie szczegółowej wiedzy może wyciągać wnioski i podejmować decyzje w oparciu o posiadaną wiedzę, działając w sposób zbliżony do procesu rozumowania człowieka, który swą umiejętność wnioskowania nabywa w wyniku studiów i praktyki. Celem takich programów jest zastąpienie pracy wielu ekspertów w danej dziedzinie poprzez przechowywanie ekspertyz wypracowanych przez nich a następnie stosowanych do rozwiązywania problemu w sposób kompletny. Rozwiązując problem podają one odpowiedź a nie dane, dostarczając wyjaśnień jak uzyskano rozwiązanie.

Wysoka użyteczność systemów ekspertowych wynika z ich właściwości, czyli poprawności systemu, uniwersalności, złożoności, autoanalizy oraz zdolności do udoskonalania bazy wiedzy.

2.1 Podstawowe wiadomości o systemach ekspertowych

System ekspertowy powinien zapewnić wysoki poziom wydawanych ekspertyz. W tym sensie możemy mówić o poprawności systemu, jeśli daje on dobre rezultaty, rozwiązuje zadania w czasie dopuszczalnym i dysponuje strategiami umożliwiającymi limitowanie wiedzy i intuicji eksperta, uzyskanej w wyniku wieloletniego doświadczenia. Jakość pracy możemy ocenić porównując wyniki działania systemu z rezultatami pracy człowieka.

Uniwersalność w przypadku systemów ekspertowych rozumiana jest jako zdolność do rozwiązywania obszernej klasy zadań z danej dziedziny. System nie powinien zawierać wielu sztywnych, wcześniej przygotowanych rozwiązań, lecz dużą liczbę reguł obejmujących dostatecznie szeroki zakres heurystyk z dziedziny problemowej. Uniwersalność rozumiana jako możliwość rozwiązania zadań z różnych dziedzin wiedzy na podstawie strukturalnego podobieństwa reguł wnioskowania jest jeszcze nieosiągalna. Tworzenie systemów zdolnych do takiego działania – można je nazwać metasystemami – jest nieuniknionym kierunkiem rozwoju badań nad sztuczną inteligencją.

Stopień komplikacji systemu ekspertowego jest w natural-

ny sposób określony przez dziedzinę, dla której jest wykonany. Ocena złożoności systemu jest możliwa na przykład przez liczbę reguł wnioskowania (dla systemu opartego o regułową bazę wiedzy), wielkość bazy danych itp. Systemy ekspertowe są klasyfikowane ze względu na liczbę reguł na trzy grupy - małe (100 – 300 reguł), średnie (300 – 2000 reguł), duże (ponad 2000 reguł).

System ekspertowy powinien uzasadnić użytkownikowi przyjęte rozwiązanie nie tylko globalnie, ale i na każdym etapie, to znaczy również każde rozwiązanie częściowe. Dokonuje się tego w ten sposób, że przegląda się drzewo rozwiązania w kierunku wstecznym, tak jakby to było jeszcze jedno zadanie wymagające ekspertyzy. W analizowaniu przez system ekspertowy własnego zachowania istotną rolę odgrywa tzw. moduł niesprzeczności. Do prowadzenia autoanalizy niezbędna jest możliwość rekonstrukcji pewnego ciągu wnioskowania. Objaśnienia są ważnym elementem pracy systemu, a ich waga rośnie wraz ze wzrostem kosztów przyjęcia błędnego rozwiązania. Do mechanizmów udoskonalających jego działanie zalicza się: kontrolera niesprzeczności nowo wprowadzanych do bazy wiedzy reguł z regułami w niej zawartymi, kontrolera zgodności reguł z nowo wprowadzanymi faktami, mechanizm oceny częstości stosowania poszczególnych reguł, mechanizm rozbudowy istniejącej bazy reguł poza zakres danej bazy wiedzy. Dwa pierwsze elementy mieszczą się w module niesprzeczności, pozostałe należy wbudować jako dodatkową strukturę uczącą [11].

System ekspertowy, jeśli ma być efektywny, powinien umożliwiać ciągłe rozszerzanie wiedzy o nowe fakty i prawa (reguły wnioskowania).

Program do rozwiązywania problemów zlecanych ekspertom, charakteryzuje się strukturą funkcjonalną, której podstawowymi elementami są [13]:

- baza wiedzy, zawierająca wiedzę potrzebną do rozwiązywanych problemów, zwaną także wiedzą dziedzinową,
- system wnioskujący, wyznaczający fakty wynikające z bazy wiedzy i z pewnego zbioru faktów początkowych, charakteryzujących problem będący przedmiotem wnioskowania.

Oprócz wymienionych elementów podstawowych system ekspertowy ma jeszcze elementy pomocnicze, do których należy:

- dynamiczna baza danych, służąca do przechowywania odpowiedzi użytkownika i wyników wnioskowania,
- edytor bazy wiedzy, służący do czytania, formułowania i modyfikowania bazy wiedzy,
- łącze użytkownika, umożliwiające korzystającemu z programu komunikowanie się z systemem wnioskującym i edytorem bazy wiedzy.

Jak już wcześniej zostało zaznaczone, istotną cechą systemu ekspertowego jest to, że baza wiedzy jest plikiem tekstowym, napisanym w sposób zrozumiały i przejrzysty. Baza ta może być tworzona, czytana i modyfikowana za pomocą edytora bazy wiedzy bez naruszania integralności systemu wnioskującego. Struktura systemu ekspertowego opiera się na założeniu, iż jego siła zależy od bazy wiedzy (jej jakości), a dopiero w następnej kolejności od dysponowanego oprogramowania.

2.2 Metody bazujące na symbolicznej reprezentacji wiedzy

Problem reprezentacji wiedzy, jest jednym z najważniejszych w dziedzinie sztucznej inteligencji, który nie został jeszcze całkowicie rozwiązany.

Wiedza w danej dziedzinie oznacza zbiór wiadomości z tejże dziedziny, wszystkie zobiektywizowane i utrwalone formy kultury umysłowej i świadomości społecznej będące wynikiem kumulowania doświadczeń i uczenia się. Wiedza inaczej jest symbolicznym opisem otaczającego nas świata rzeczywistego.

Gromadzenie wiedzy jest niejednokrotnie najtrudniejszym zadaniem podczas tworzenia systemu ekspertowego, najczęściej wykonywanym przez inżyniera wiedzy, który uzyskuje niezbędne informacje od eksperta oraz z innych dostępnych źródeł i umieszcza je odpowiednio przetworzone w bazie wiedzy.

Można wyróżnić dwa typy symbolicznej reprezentacji wiedzy – reprezentacja proceduralna, polegająca na określeniu zbioru procedur, działanie których reprezentuje wiedzę o dziedzinie oraz reprezentacja deklaratywna, polegająca na określeniu zbioru specyficznych dla rozpatrywanej dziedziny faktów, stwierdzeń, reguł.

Sposób reprezentacji wiedzy w bazie wiedzy jest ściśle zależny od charakteru systemu ekspertowego i zadań jakie są przed nim stawiane. Najczęściej spotykanymi technikami organizowania baz wiedzy są [12]:

- metody bazujące na bezpośrednim zastosowaniu logiki (rachunek zdań, rachunek predykatów),
- metody wykorzystujące zapis stwierdzeń,
- metody wykorzystujące systemy regułowe (wektory wiedzy),
- metody z wykorzystaniem sieci semantycznych,
- metody oparte na ramach,
- metody używające modeli obliczeniowych.

Rachunek zdań to jedna z podstawowych koncepcji mechanizmów wnioskowania wywodząca się z logiki dwuwartościowej. Opis cech otaczającego nas świata formułuje się w postaci zdań. Zdanie jest zdaniem prawdziwym jeśli jego wartość logiczna wynosi 1, natomiast jest fałszywe jeśli ma wartość logiczną 0. Zdania oznacza się symbolami, np. A,B... i mogą być one łączone za pomocą spójników logicznych tworząc wyrażenia logiczne tzw. formuły. Formuły dające zdanie prawdziwe niezależnie od wartości logicznych zmiennych zdaniowych nazywamy prawami rachunku zdań (tautologiami).

Bazy wiedzy oparte na ***logice***, mimo swej modularności, deklaratywności i nieproceduralności, są trudne do przetwarzania, szybko następuje w nich eksplozja kombinatoryczna, czyli lawinowy i niekontrolowany rozrost bazy wiedzy o fakty będące powieleniem istniejących już informacji lub powstaniem niepożądanych struktur.

Ważnym narzędziem w metodach reprezentacji wiedzy

jest ***rachunek predykatów***, który stanowi podstawę programowania w logice. Rachunek predykatów jest rozszerzeniem rachunku zdań przez wprowadzenie kwantyfikatorów: „dla każdego" oraz „istnieje takie że". Wyrażenie W(x), w którym występuje zmienna x i które staje się zdaniem prawdziwym lub fałszywym, gdy w miejsce x podstawimy wartość zmiennej x, nazywa się funkcją zdaniową albo predykatem. Predykat składa się z nazwy i dowolnej liczby argumentów, które są nazywane termami. Termami mogą być stałe (symbole) alfanumeryczne, jak też numeryczne i zmienne oraz wyrażenia [12].

Zaletą predykatów są proste i zrozumiałe interpretacje wyrażania zdań. Trzeba wziąć pod uwagę fakt, iż nie wszystkie pojęcia o otaczającej nas rzeczywistości dają się reprezentować w logice, jednakże obecnie trudno byłoby wskazać metodę opisu, która sprostałaby wszystkim wymaganiom. Rachunek predykatów dobrze opisuje podstawy matematyki, która to z kolei wystarcza do opisu formalizmów większości współczesnych dyscyplin naukowych. Istotną zaletą teorii logicznych jest ich formalność, czyli możliwość ścisłego i jednoznacznego opisu ich konstrukcji i mechanizmów, co przy reprezentowaniu wielu problemów odgrywa znaczącą rolę.

Z kolei ***stwierdzenia*** są jednym z głównych elementów baz wiedzy, dotyczą one takich zagadnień jak zdarzenia, zjawiska, objawy, czynności i są zapisywane w postaci uporządkowanej trójki – (<OBIEKT>,<ATRYBUT>,<WARTOŚĆ>). W celu uproszczenia zapisów stwierdzeń stosuje się słowniki nazw obiektów i atrybutów oraz ich wartości, dzięki temu uzyskuje się oszczędniejszy zapis, który zajmuje mniej miejsca w pamięci komputera.

W wypadku tzw. stwierdzeń przybliżonych do uporządkowanej trójki dodaje się czwarty człon, który jest współczynnikiem pewności (<CF> - Certainty Factor). CF jest najczęściej liczbą z przedziału [-1,1] lub [-1,0] bądź [0,1], która określa stopień pewności stwierdzenia np. dla drugiego z wymienionych przedziałów CF=1 oznacza, że stwierdzenie jest w pełni prawdziwe, CF=0 to stwierdzenie fałszywe. Nie sformalizowano metod określenia stopni pewności, dlatego wyznacza się je subiektywnie. Należy zaznaczyć, że w reprezentacji wiedzy za pomocą stwierdzeń występują trudności negowania stwierdzeń w przypadku atrybutów wielowartościowych.

Istotną pozycję wśród metod reprezentacji wiedzy zajmują ***metody oparte na regułach***, gdyż zbiór stwierdzeń nie jest wystarczający do opisania jakieś dziedziny wiedzy. Ogólna postać reguły może być wyrażona jako - JEŚLI przesłanka TO konkluzja (działanie). Przesłanka może zawierać większą liczbę stwierdzeń połączonych funktorami logicznymi. Ważną zaletą tej metody jest to, iż nie powoduje ona niekontrolowanego rozrostu bazy wiedzy o nadmiarowe tautologie. Baza wiedzy w tym przypadku zawiera zbiór reguł. ***Podejście to umożliwia uzyskanie dużej modularności bazy wiedzy i dlatego jest stosowane w większości systemów ekspertowych***.

Praktycznie działające systemy oparte na regułach mogą zawierać reguły charakteryzowane stopniami pewności.

Zbiór reguł to pewnego rodzaju sieć stwierdzeń, ponieważ z prawdziwości jednego stwierdzenia mogą wynikać inne.

Należy nadmienić, iż reguły o jednym wniosku noszą nazwę klauzul Horna. Zaletą stosowania reguł w tej postaci jest uproszczenie automatyzacji wnioskowania, a więc budowy systemu wnioskującego, czyniąc go zarazem efektywnym. Prostota klauzul Horna czyni je zrozumiałymi i przejrzystymi dla użytkownika nie będącego informatykiem.

Pewnego rodzaju uogólnieniem reguł, w wyniku którego otrzymujemy zapis w postaci wektorowej, są ***wektory wiedzy***. W podejściu tym najpierw zapisujemy daną bazę reguł w tradycyjny sposób, uwzględniając to by poszczególne reguły zawierały jednakową liczbę warunków i wniosków. W kolejnym kroku dokonujemy kodowania poszczególnych członów reguł z wykorzystaniem symboli (TAK, NIE, nie występuje). W rezultacie zamiast pisać pełną reprezentację poszczególnych reguł, otrzymujemy bardzo zwarty opis w postaci wektorów, zawierających trzy wymienione symbole.

Wektory wiedzy są wygodne do weryfikacji poprawności bazy wiedzy a mając postać wektorową łatwo jest przejść na opis zawierający pełną treść reguł.

Następnym modelem stosowanym przy reprezentowaniu wiedzy są ***sieci semantyczne***. Baza wiedzy stanowi zbiór stwierdzeń i relacji między nimi, stąd posługując się tymi pojęciami można stworzyć tzw. sieć stwierdzeń – w formie grafu, gdzie węzłami są stwierdzenia, a gałęziami relacje. Istnieje możliwość przypisania węzłom (podobnie jak gałęziom) wag, które określają stopień przekonania o słuszności tych stwierdzeń.

Uogólnieniem sieci stwierdzeń są sieci semantyczne lub asocjacyjne, a polega to na przyjęciu założenia, że węzły odpowiadają kompletnym opisom pojęć lub obiektów i nie są wyłącznie stwierdzeniami. Model asocjacyjny to taki model, którego jedne terminy są wyjaśniane przez inne terminy, tworząc pewną strukturę powiązań. Struktura ta może być zamknięta, a grafy mogą być skierowane. Wnioskowanie odpowiada poruszaniu się po grafie, gdyż sieć semantyczna jest pewnego rodzaju logiką, gdzie relacje między obiektami są przedstawione w postaci rysunku.

Sieci semantyczne wiążą się zazwyczaj z ***ramami*** lub z regułami, gdyż problemem jest tutaj określenie, czy węzły sieci oznaczają jeden obiekt czy klasę obiektów. Ramy w tym ujęciu odpowiadają obiektom i opisują ich strukturę wewnętrzną, sieć semantyczna natomiast odpowiada relacjom między ramami.

Reprezentacja wiedzy za pomocą ram umożliwia deklaratywną i proceduralną reprezentację wiedzy oraz organizuje bazę wiedzy w taki sposób, że reguły będące reprezentacją wiedzy danej dziedziny są wyraźnie oddzielone od reguł niezbędnych do poprawnego działania systemu ekspertowego.

Możliwość grupowania informacji dotyczących wybranego fragmentu wiedzy w postaci jednej ramy upraszcza późniejszą weryfikację i modyfikacje bazy wiedzy. Rama jest zatem strukturą danych opisującą dany obiekt, zawierającą wszystkie typowe i oczekiwane informacje przy-

puszczenia o tym obiekcie.
Pomysł reprezentacji wiedzy w postaci ram został oparty na podstawie analizy sposobu zachowania człowieka. Człowiek znajdujący się w nowej sytuacji i nowym otoczeniu, ale mający o tej dziedzinie wcześniejsze wyobrażenia, wydobywa z pamięci określoną strukturę, czyli ramę, i konfrontuje tę sytuację z wiedzą zawartą w ramie. Wykreowanie nowej ramy związane jest z zetknięciem się człowieka z całkowicie nowym obiektem, a więc próbą zapamiętania go i wprowadzenia jego nazwy.

2.3 Metody wnioskowania

Metody wnioskowania decydują o tym, w jaki sposób zachodzi proces myślenia.
Wyróżnia się trzy podstawowe typy wnioskowania: w przód (progresywne), wstecz (regresywne) i mieszane [12].
Metoda przeszukiwania poprzez stosowanie odpowiednich heurystyk jest często stosowaną techniką, gdyż dla większości problemów trudno jest z góry określić ciąg czynności prowadzących do rozwiązania.
Osobną grupę stanowią techniki wnioskowania wykorzystujące wiedzę niepewną, mamy na myśli przede wszystkim wnioskowanie rozmyte.

2.3.1 Wnioskowanie progresywne, regresywne i mieszane

Idea ***wnioskowania w przód*** jest następująca – na podstawie dostępnych reguł i faktów należy generować nowe fakty tak długo, aż wśród wygenerowanych faktów znajdzie się postawiony cel (hipoteza).
Cechą charakterystyczną tego typu wnioskowania jest możliwość powiększania się bazy faktów, co w pewnych sytuacjach może zostać uznane jako jego wada, gdyż pamięć operacyjna komputera może zostać całkowicie zapełniona. Jednocześnie z innego punktu widzenia, postępowanie takie umożliwia w przypadku baz wiedzy o niewielkiej liczbie faktów, zwiększenie jej liczby, a tym samym przyspieszenie procesu sprawdzania postawionej hipotezy [12].
Natomiast ***wnioskowanie wstecz*** przebiega w odwrotną stronę niż wnioskowanie w przód. Polega ono na wykazaniu prawdziwości hipotezy głównej na podstawie prawdziwości przesłanek. W sytuacji, gdy nie wiemy, czy jakaś przesłanka jest prawdziwa, to traktujemy tę przesłankę jako nową hipotezę i próbujemy ją wykazać. Jeśli postępując w ten sposób znajdziemy wreszcie regułę, której wszystkie przesłanki są prawdziwe, to konkluzja tej reguły jest prawdziwa. Na podstawie tej konkluzji dowodzi się następną regułę, której przesłanka nie była poprzednio znana. Postawiona hipoteza jest prawdziwa, jeśli wszystkie rozważane przesłanki dadzą się wykazać [12].
Zasadniczą cechą, która odróżnia wnioskowanie wstecz od wnioskowania w przód jest mniejsza liczba generowanych nowych faktów oraz niemożność równoczesnego dowodzenia kilku hipotez. Przy wnioskowaniu wstecz czas oczekiwania na rozwiązanie jest w wielu przypadkach dużo krótszy niż przy wnioskowaniu w przód, dlatego też tego typu wnioskowanie w typowych zastosowaniach jest efektywniejsze i bardziej rozpowszechnione.
Wnioskowanie mieszane pozbawione jest niektórych wad wspomnianych metod, gdyż jest kompromisem między wnioskowaniem w przód i wstecz.
Strategia tego wnioskowania opiera się na wykorzystaniu ogólnych reguł, tzw. meta-reguł stanowiących metawiedzę, dzięki czemu program zarządzający dokonuje odpowiedniego przełączania między poszczególnymi rodzajami wnioskowania. W zależności od sytuacji system może automatycznie dobierać najbardziej odpowiedni sposób wnioskowania, gdyż w meta-regułach są zawarte wskazania dotyczące priorytetów wyboru rodzaju wnioskowania. System oparty na wnioskowaniu mieszanym działa tak, jakby można było w nim wyróżnić dwie maszyny wnioskujące (progresywną i regresywną), dlatego też poza wczytaniem przez system bazy wiedzy należy wczytać także zbiór zawierający metareguły. Wiedza zapisana w metaregułach może preferować jeden z rodzajów wnioskowania, co uzyskuje się na przykład poprzez podział bazy wiedzy na dwie części – reguły związane z wnioskowaniem wstecz oraz reguły związane z wnioskowaniem w przód.
Główną zaletą wnioskowania mieszanego jest skrócenie czasu potrzebnego na uzyskanie rozwiązania oraz nie występuje tutaj zagrożenie zajęcia całej pamięci operacyjnej komputera.
Trudność natomiast sprawia pozyskanie metawiedzy – źle dobrane metareguły mogą spowolnić pracę systemu lub nawet uczynić ją nieefektywną.

2.3.2 Strategie przeszukiwania, heurystyki

Wybór sposobu postępowania prowadzącego do uzyskania określonych wyników odgrywa istotną rolę podczas rozwiązywania problemu.
Metoda przeszukiwania jest często stosowaną techniką wnioskowania, gdyż dla większości problemów trudno jest z góry określić ciąg czynności prowadzących do rozwiązania, muszą one być określone przez systematyczne analizowanie kolejnych alternatyw.
Zaletą przeszukiwania jest łatwość formułowania zadań – wymagane jest określenie zbioru stanów przestrzeni rozwiązywanego problemu, zbioru operatorów przekształcających te stany, stanu początkowego i zbioru stanów końcowych. Określenie ciągu operatorów przekształcających stan początkowy w stan końcowy to rozwiązanie danego problemu.
W zagadnieniach przeszukiwania korzysta się z pewnych algorytmów, czyli strategii realizujących poszczególne metody przeszukiwań. Strategie przeszukiwania mówią, w jaki sposób maszyna wnioskująca ma sprawdzać prawdziwość kolejnych stanów.

W grupie tych strategii znajdują się metody nie wykorzystujące informacji o dziedzinie rozwiązywanego problemu, zwane metodami ślepymi, a także metody ściśle dopasowane do danego problemu, wykorzystujące tzw. metody heurystyczne.

Koncepcja ***strategii heurystycznych*** wywodzi się ze spostrzeżenia, że dla większości problemów przestrzeń stanów zawiera pewne dodatkowe informacje. Koszt wyznaczenia tych informacji jest niewielki, a umożliwiają one wybieranie najlepszych kierunków przeszukiwań, podając proste kryterium ich wyboru [12].

Heurystyki pomagają zoptymalizować poszukiwania rozwiązań, pozwalają pominąć ścieżki, które nie rokują nadziei na odnalezienie rozwiązania, skracają czas dochodzenia do wyniku. Heurystyczne przeszukiwanie jest procesem poszukiwania żądanego stanu albo inaczej podgrafu spełniającego zadane warunki, posługując się różnymi środkami, takimi jak analogie, uproszczenia, intuicje, których celem jest ograniczenie zbioru przeszukiwanych stanów, a których użyteczność nie jest do końca znana. Poszukiwanie pożądanego stanu odbywa się często w sposób subiektywny, oparty na regułach wypracowanych przez ekspertów. Dlatego też metody heurystyczne są wykorzystywane wtedy, gdy nie ma algorytmu lub tradycyjne algorytmy wyznaczają niezadowalające rozwiązanie albo nie dają gwarancji rozwiązania zadania. Heurystyka nie daje pewności znalezienia rozwiązania, jednakże właściwie dobrana powinna wyznaczać najlepsze wyniki osiągane w żądanym czasie. Jest to praktyczna strategia poprawiająca efektywność rozwiązywania złożonych problemów, prowadząc do celu wzdłuż najkrótszej, najbardziej prawdopodobnej drogi [12].

Strategia „***najpierw najlepszy***" wykorzystuje pewną informację heurystyczną związaną z rozwiązywanym problemem do zminimalizowania kosztów przeszukiwania. Stosuje się pewną funkcję heurystyczną, która wyraża ocenę węzła ze względu na zbieżność, czyli osiągnięcie celu, najmniejszego kosztu drogi, najmniejszej złożoności obliczeniowej procesu przeszukiwania. Następnie do rozszerzania wybierany jest „najlepszy" węzeł spośród wszystkich węzłów rozpatrywanych do tej pory, niezależnie od ich położenia w grafie. Rozszerzenie węzłów jest dokonywane przez ekspansję, czyli generowanie wszystkich potomków.

Inną popularną metodą heurystycznego przeszukiwania jest ***strategia A****, której celem jest wyznaczenie najtańszej drogi w grafie. Wyrażenie f(w)=h(w)+g(w) to tzw. funkcja heurystyczna, która oznacza, że dla danego węzła w jest wyznaczana w sposób heurystyczny estymacja h(w) kosztu drogi łączącej węzeł w z węzłem celu, a następnie wyznacza się dla węzła w koszt drogi łączącej węzeł początkowy p z węzłem w, co reprezentuje składnik g(w).

Najbardziej znaną ***strategią ślepą*** jest ***strategia w głąb***, a nazwa jej podkreśla kolejność przeszukiwania grafu. Droga w grafie jest wyznaczana dopóty, dopóki jej ostatni element nie zostanie określony jako węzeł celu lub końcowy. Przeszukiwanie jest zawsze prowadzone od ostatnio sprawdzanego węzła, którego nie wszystkie gałęzie były jeszcze rozwijane. Główną operacją strategii w głąb jest ekspansja węzłów, tzn. generowanie wszystkich ich potomków.

Strategia ta może być nieskuteczna, gdy zostanie zastosowana do grafów o dużej głębokości, dlatego też omawiana metoda jest zazwyczaj uzupełniana mechanizmem kontroli ograniczenia głębokości. W sytuacji, gdy głębokość węzła przekracza ograniczenie lub węzeł spełnia kryterium końcowe, następuje powrót. Najważniejszą cechą strategii w głąb jest badanie kolejnych węzłów wzdłuż jednej ścieżki, z tego powodu strategia ta jest naturalna dla tych problemów i grafów, w których ocena właściwości węzłów zależy ściśle od oceny właściwości ich rodziców. W strategii tej dość łatwo można oszacować wymagania pamięciowe, gdyż w pamięci przechowywane są węzły z aktualnie badanej ścieżki grafu.

Kolejną strategią jest ***strategia z powracaniem*** i jest pewnego rodzaju modyfikacją algorytmu przeszukiwania w głąb. Główna różnica polega na tym, że ekspansja badanego węzła jest zastąpiona jego rozszerzeniem, czyli generowaniem jednego potomka. W sytuacji, gdy ten nowy węzeł nie spełnia kryterium celu lub końcowego, to jest dalej rozszerzany. Jeżeli po kolejnych rozszerzeniach otrzymany węzeł spełnia kryterium końca przeszukiwanego grafu lub nie można dla niego wygenerować nowego potomka, to następuje powrót do najbliższego przodka, dla którego jest możliwe wygenerowanie potomków.

W porównaniu do poprzednio omawianej strategii, ***strategia z powracaniem*** charakteryzuje się oszczędnością pamięci. Strategia ta gwarantuje, że po jej zakończeniu wszystkie wygenerowane węzły są przetestowane, czego nie zapewnia strategia w głąb, gdzie część węzłów otrzymanych w wyniku kolejnych ekspansji może w tym przypadku odbywać się kosztem komplikacji strategii. Oszczędność pamięci w tym przypadku odbywa się kosztem komplikacji strategii.

Natomiast ***strategia wszerz***, badając kolejno poziomy, przyznaje wyższy priorytet węzłom o mniejszej głębokości. Algorytm wszerz wyznacza węzeł celu o najmniejszej głębokości w porównaniu z innymi węzłami celu. Główną operacją strategii wszerz jest ekspansja węzłów. Strategia ta daje gwarancję, że dla lokalnie skończonych grafów, czyli takich, w których każdy węzeł ma skończoną liczbę potomków, osiągnie się węzeł celu, jeśli istnieje.

Strategia ta ma bardzo pożądaną cechę zbieżności i jako pierwsze wyznacza rozwiązanie optymalne pod względem długości ścieżki rozwiązania. W strategii tej występują duże wymagania pamięciowe, gdyż analizowane są wszystkie węzły o głębokości mniejszej od głębokości wyznaczonego węzła celu, zamiast jednej ścieżki w pamięci przechowywane są wszystkie węzły o danej głębokości przed wygenerowaniem jakiegokolwiek węzła o głębokości o jeden większej.

Innym przykładem metod przeszukiwania jest ***strategia zachłanna***. Główną operacją tej strategii jest ekspansja węzłów, po jej wykonaniu są badane nowe węzły i najbardziej obiecujący jest wybierany do dalszej ekspansji. W strategii zachłannej niemożliwe są powroty do żadnego

przodka aktualnie badanego węzła, gdyż strategia ta wykorzystuje lokalną optymalizację. Strategia ta odznacza się prostym algorytmem obliczeniowym, jednakże jej poważną wadą jest nieodwracalność, czyli brak możliwości powrotu do kierunków przeszukiwania, które na pewnym etapie były lokalnie gorsze. Taka właściwość może implikować sytuację w której badana jest droga prowadząca do węzła końcowego nie spełniającego kryterium celu lub do penetrowania drogi nieskończonej.

3. Wykorzystanie systemu ekspertowego w procesie decyzyjnym dotyczącym doboru kadry

Poniżej omówiony zostanie skorupowy system ekspertowy (system pozbawiony własnej wiedzy dziedzinowej) przybliżony, modelujący niejednoznaczność wiedzy za pomocą współczynników pewności, opisany przez Antoniego Niederlińskiego, który został wykorzystany do analizy problemu doboru kadry w przedsiębiorstwie. Reguły do bazy wiedzy były dobrane na podstawie teorii zarządzania w przedsiębiorstwie.

3.1 Budowa systemu ekspertowego-skorupowego opartego o wnioskowanie elementarne przybliżone w przód

Istotnym elementem systemu jest struktura bazy wiedzy, która ma duże znaczenie dla żywotności i przyjazności systemu ekspertowego, należy projektować ją więc tak, aby składała się z plików tekstowych - bazy reguł, bazy ograniczeń, bazy rad. Nazwy wymienionych baz są pierwszymi dwoma literami odpowiednich plików tekstowych.
Baza wiedzy, która jest integralną częścią systemu ekspertowego ma tu charakter indywidualny. Podstawową częścią bazy wiedzy są reguły.
Reguły bazy reguł mogą się zagnieżdżać, oznacza to, że wnioski niektórych reguł są warunkami innych reguł. Taka własność bazy reguł sprawia, że nie o wszystkie warunki powinien się system ekspertowy pytać użytkownika. Dlatego zbiór wszystkich warunków bazy reguł dzieli się na warunki dopytywalne, czyli takie, których wartość logiczna musi być określana przez użytkownika w odpowiedzi na pytanie zadane przez system wnioskujący, oraz warunki niedopytywalne, których wartość logiczna wynika z odpowiadających im reguł i wartości logicznej odpowiednich warunków dopytywalnych.
Ze względu na potrzebę stosowania różnych mechanizmów wnioskowania można bazy reguł klasyfikować w zależności od struktury zagnieżdżania reguł oraz pewności reguł.
W zależności od struktury zagnieżdżania reguł wyróżnia się bazy reguł elementarne, których warunki niedopytywalne nie mogą występować w postaci zanegowanej, oraz bazy reguł rozwinięte, mogące zawierać zanegowane warunki niedopytywalne.
Natomiast w zależności od pewności reguł rozróżnia się bazy reguł dokładne, których reguły są prawdziwe, a warunki i wnioski mogą być albo prawdą, albo nieprawdą, oraz bazy reguł przybliżone, których reguły, warunki i wnioski mogą mieć różne stopnie pewności.
Podczas konstrukcji bazy reguł mogą się w niej pojawić sprzeczności, które powinny zostać usunięte przed przystąpieniem do wnioskowania. Obecność sprzeczności może doprowadzić do niekończących się pętli a w konsekwencji do zawieszania się systemu wnioskującego.
W bazie reguł mogą się także pojawić nadmiarowości, które objawiają się jako niepotrzebne reguły i warunki, czyli takie które wyrażają to samo, co inne reguły i warunki oraz reguły, które zawierają dla pewnych wniosków bardziej złożone zestawy warunków aniżeli inne reguły dla tych samych wniosków. Nadmiarowość może być źródłem niedokładnych wyników wnioskowania, poza tym powiększa rozmiary bazy reguł i wydłuża czas wnioskowania.
Często baza reguł może nie mieć własnych faktów, lecz korzystać wyłącznie z faktów deklarowanych przez użytkownika – jednorazowo przed rozpoczęciem wnioskowania, lub kolejno w trakcie wnioskowania. Fakty takie są przechowywane w dynamicznej bazie danych.
Kolejną składową bazy wiedzy jest baza ograniczeń, która zawiera zbiory warunków dopytywalnych wykluczających się. W przeciwieństwie do bazy reguł, która jest niezbędną częścią bazy wiedzy, bazy ograniczeń może w ogóle nie być. Jednakże konstrukcja bardziej inteligentnego systemu ekspertowego wymaga wprowadzenia do bazy wiedzy informacji o zbiorach warunków dopytywalnych. Dzięki takiemu postępowaniu, system w przypadku uznania za prawdę jednego z kilku wykluczających się warunków lub w przypadku uznania za nieprawdę jednego z dwóch dychotomicznych warunków nie powinien już pytać o pozostałe warunki lub pozostały warunek, lecz automatycznie określić ich wartość logiczną.
Kolejną składową bazy wiedzy jest baza rad, która jest plikiem tekstowym zawierającym uporządkowane pary (numer_reguły, nazwa_pliku_tekstowego_rady_dla_reguły). Baza rad jest swego rodzaju katalogiem plików tekstowych rad dla danej bazy reguł.
Pliki rad są plikami tekstowymi rad, przyporządkowanymi poszczególnym regułom bazy reguł. Baza rad i pliki rad nie są niezbędnymi składowymi bazy wiedzy, bez nich system będzie nadal wnioskował, nie udzielając jednak użytkownikowi żadnych wskazówek.
Dane, na podstawie których dokonywane jest wnioskowanie w systemie ekspertowym, będące albo deklaracjami użytkownika albo wnioskami wynikłymi z dotychczasowych wnioskowań, są przechowywane w dynamicznej bazie danych. Ze wspomnianej bazy danych odczytywane są informacje przez system wnioskujący podczas testowania dalszych reguł. W ten sposób unika się powtórnego pytania użytkownika o prawdziwość warunku wcześniej przez niego uznanego lub nie uznanego za fakt oraz zapewnia się jednolitą obsługę danych przez system wnioskujący, niezależnie od ich pochodzenia.

Kończąc omawianie poszczególnych składowych systemu ekspertowego należy wspomnieć także o interfejsie użytkownika, który jest jego istotnym elementem. Dzięki łączu użytkownika możliwe jest ładowanie wybranej bazy wiedzy, wprowadzenie danych potrzebnych do wnioskowania, edytowanie wybranej bazy wiedzy, tworzenie nowej bazy wiedzy, kasowanie istniejącej bazy wiedzy, czytanie bądź kasowanie raportów wnioskowania.

Współczynniki pewności służą do oceny stopnia pewności warunków dopytywalnych, reguł oraz wniosków wyprowadzonych z owych niepewnych reguł i niepewnych warunków dopytywalnych. Wnioskowanie przy użyciu współczynników pewności opiera się na założeniu, iż każdemu wnioskowi przyporządkowany jest odpowiedni współczynnik CF, charakteryzujący pewność tego, że warunek jest lub nie jest prawdziwy. Przykładowo warunek, którego prawdziwość jest całkowicie pewna oznaczany jest jako CF=1, z kolei CF=-1 dotyczy przekonania o nieprawdziwości warunku.

Współczynniki pewności warunków dopytywalnych są deklarowane przez użytkownika przed lub w trakcie wnioskowania, często są wynikiem uśrednienia opinii szerokiego grona ekspertów.

Prowadzone są również intensywne badania teoretyczne i numeryczne, zmierzające do opracowania odpowiedniej metodologii wyznaczania współczynników pewności.

Współczynniki pewności warunków niedopytywalnych są wyznaczane w drodze wnioskowania. Wprowadzenie ujemnych wartości współczynników pozwala uwzględnić warunki sprzyjające i niesprzyjające wnioskowi, czyli takie, które powiększają bądź zmniejszają współczynnik pewności wniosku.

Tylko jeden warunek z listy warunków wykluczających się zawartej w bazie ograniczeń może mieć CF=1; dla pozostałych CF=-1.

Reguły zapisywane są w bazie reguł w postaci klauzul-faktów:

Reguła(Nr_reguły, Wniosek,[Warunek_1,...,Warunek_n])

Również każdej regule przyporządkowany jest współczynnik pewności. Współczynnik ten jest swego rodzaju wzmocnieniem określającym wpływ pewności warunków reguły na pewność wniosku reguły. I tak na przykład CF=0,5 oznacza regułę, której warunki w połowie wzmacniają pewność wniosku swoją pewnością. Współczynniki pewności reguł są elementami bazy reguł i są one zapisywane w postaci:

Nr_reguły Warunek_1, Warunek_2,...Warunek_n-CF Wniosek.

Kolejna zasada dotyczy współczynnika pewności koniunkcji warunków znajdujących się w liście warunków reguły, który jest określany jako najmniejszy spośród współczynników pewności koniugowanych warunków:

CF(A,B,C,...) = min(CF(A),CF(B),CF(C),...)

Współczynnik pewności wniosku reguły jest iloczynem współczynnika pewności reguły i współczynnika pewności koniunkcji warunków tej reguły. Dla reguły niepewnej:

A,B,C,...-CF_reguły W

dla której jest:

CF(A,B,C,...) = min(CF(A),CF(B),CF(C),...) = CF_wniosków

będzie:

CF(W) = CF_reguły*CF_wniosków

Współczynnik pewności sumy logicznej jednakowych wniosków należy wyznaczyć wtedy, jeżeli baza reguł ma szereg reguł dla tego samego wniosku. Współczynnik ten dla dwóch jednakowych wniosków równy jest sumie współczynników pewności tych wniosków, zmniejszonej o ich iloczyn, zakładając, że przynajmniej jeden z nich ma nieujemną wartość CF:

CF(Wniosek)=CF_1(Wniosek)+CF_2(Wniosek)-CF_1(Wniosek)*CF_2(Wniosek).

Współczynnik pewności sumy logicznej dwóch jednakowych wniosków, z których obydwa mają niedodatnie wartości CF, jest równy sumie współczynników pewności tych wniosków, powiększonej o ich iloczyn:

CF(Wniosek) = CF_1(Wniosek)+CF_2(Wniosek)
+CF_1(Wniosek)*CF_2(Wniosek).

W przypadku większej liczby reguł z jednakowymi wnioskami postępujemy podobnie, po wyznaczeniu wartości dla dwóch pierwszych reguł, łączymy ją z wartością dla trzeciej reguły, itd.

W trakcie konstrukcji elementarnej przybliżonej bazy wiedzy mogą się do niej wkraść sprzeczności. Rozróżnia się następujące rodzaje sprzeczności [12]:

- Sprzeczności typu SEP1. Źródłem sprzeczności typu SEP1 jest tylko baza reguł, cyfra 1 oznacza, że sprzeczność jest generowana w obrębie jednej bazy. Sprzeczności typu SEP1 są typu zewnętrznego, czyli wniosek reguły jest tożsamy z jednym z jej warunków.

Rozróżnia się następujące przypadki sprzeczności zewnętrznych typu SEP1:

- Reguła jest zewnętrznie SEP1-samosprzeczna, jeżeli jednym z jej warunków jest jej wniosek:

X,Y,Z-CF Z

- Reguła 1 jest zewnętrznie bezpośrednio SEP1-sprzeczna z regułą 2:

X,Y,Z-CF1 W
P,U,W-CF2 Z

Zastąpienie np. warunku X reguły 1 czyni regułę 2 zewnętrznie SEP1-samosprzeczną.

1.3. Reguła 1 jest zewnętrznie pośrednio SEP1-sprzeczna z regułą 2, jeżeli podstawienie reguły 2 do innej reguły, tej zaś do jeszcze innej itd., doprowadza do reguły bezpośrednio SEP1-sprzecznej z regułą 1.

Wartości CF we wszystkich przypadkach nie wpływają na konkluzję o sprzeczności i są utrzymywane na pierwotnych wartościach przy spłaszczaniu reguł.

2. Sprzeczności typu SEP2. Źródłem sprzeczności typu SEP2 jest interakcja bazy wiedzy i bazy ograniczeń. Sprzecznością typu SEP2 jest występowanie reguły o warunkach wykluczających się w wyniku ograniczenia istniejącego w bazie ograniczeń np.:

A,B,C-CF1 W, przy liście warunków wykluczających się w bazie ograniczeń:(A,C).

W celu wykrycia tej sprzeczności, dokonuje się spłaszczenia reguł, tzn. wyraża wszystkie reguły za pomocą warunków dopytywalnych. Następnie testuje się owe warunki dopytywalne na obecność par warunków wykluczających się a danych przez bazę ograniczeń.

W elementarnych przybliżonych bazach wiedzy mogą występować nadmiarowości, które dotyczą tylko reguł o tym samym wniosku i jednakowych współczynnikach pewności np.:

A,B,C-0,3 W

A,B-0,3 W

Jednakże należy mówić o nadmiarowości niepewnej, gdyż na podstawie przytoczonego powyżej przykładu, twórca bazy reguł chciał uzyskać mały współczynnik pewności dla wniosku W w przypadku małego współczynnika pewności dla warunku C. Dlatego też występują następujące rodzaje nadmiarowości niepewnej [12]:

3. Nadmiarowości niepewne typu NEP1, których źródłem może być tylko elementarna dokładna baza reguł. Rozróżnia się:

3.1. Nadmiarowości niepewne typu NEP1.1, których istotą jest występowanie reguł o jednakowych wnioskach i jednakowych warunkach.

3.2. Nadmiarowością niepewną typu NEP1.2 jest występowanie reguł subsumowanych o jednakowym CF. Nadmiarowość ta występuje w przypadku, gdy jedna reguła jest subsumowana (zawarta) w innej regule.

3.3. Nadmiarowość niepewna typu NEP2 to taka, której źródłem jest interakcja bazy reguł i bazy ograniczeń np.:

A,B,C-CF W

A,B,D-CF W

A,B,E-CF W, baza ograniczeń(C,D,E).

Łatwo zauważyć, że wszystkie trzy reguły można zastąpić jedną: A,B-CF W.

Celem wnioskowania elementarnego przybliżonego w przód (typ wnioskowania systemu szkieletowego wykorzystanego do konstrukcji omawianego w pracy systemu ekspertowego) jest wyznaczenie współczynników pewności dla wszystkich wniosków elementarnej przybliżonej bazy reguł, z uwzględnieniem odpowiadającej jej bazy ograniczeń oraz współczynników pewności warunków dopytywalnych, zadeklarowanych przez użytkownika. W wyniku wielokrotnego testowania elementarnego przybliżonego wszystkich reguł, w kolejności występowania ich w bazie reguł, wyznaczane są współczynniki pewności dla wszystkich wniosków.

Testowaniem elementarnym przybliżonym nazywa się wyznaczenie współczynnika pewności wniosku reguły na podstawie znajomości współczynnika pewności reguły oraz współczynników pewności jej warunków i wynikającą stąd aktualizację dynamicznych baz danych. Kolejne testowania elementarne przybliżone wszystkich reguł nazywa się cyklem testowania elementarnego przybliżonego. Podczas testowania elementarnego przybliżonego mogą pojawić się różne wyniki.

Jeżeli reguła ma warunek niedopytywalny o nieokreślonym współczynniku pewności, reguła jest nieokreślona i tymczasowo pomijana, wówczas następna reguła jest testowana. Może się jednak okazać, że przy kolejnym testowaniu, współczynnik pewności niedopytywalnego warunku tej reguły został w międzyczasie określony i można wreszcie wyznaczyć współczynnik pewności dla wniosku tej reguły.

Jeżeli natomiast wszystkie warunki reguły mają określone współczynniki pewności, to reguła jest spełniona i dla jej wniosku można wyznaczyć współczynnik pewności.

Z kolei, gdy kilka reguł ma ten sam wniosek, to należy dla wszystkich tych reguł wyznaczyć współczynniki pewności wniosków i zastosować je do wyznaczenia współczynnika pewności sumy logicznej jednakowych wniosków.

3.2 Zastosowanie - symulacja oceny kandydatów ubiegających się o stanowisko w firmie

Przykładowy test został przeprowadzony dla wersji skróconej bazy wiedzy w trzech wariantach:

Tabela 1. Wyniki testów dla kandydatów na poszczególne szczeble zarządzania.

	TEST MAKSIMUM			**TEST POŚREDNI**			**TEST MINIMUM**		
Warunki dopytywalne i **niedopytywalne dla kierownictwa szczebla:**	Naczelnego	Średniego	Niższego	Naczelnego	Średniego	Niższego	Naczelnego	Średniego	Niższego
odpowiednie kwalifikacje	1	1	1	-1	-0.8	1	-1	-1	-1
doświadczenie	1	1	1	-1	-0.9	1	-1	-1	-1
wyjazdy zagraniczne	1	1	1	0.6	0.6	0.6	-1	-1	-1
konferencje, szkolenia	1	1	1	-1	-0.9	1	-1	-1	-1
rozwijanie zainteresowań	1	1	1	0.5	0.5	0.5	-1	-1	-1
studia zagraniczne	1	1	1	-1	-1	-1	-1	-1	-1
znajomość najnowszych publikacji, narzędzi pracy	1	1	1	-1	-0.8	1	-1	-1	-1
podnoszenie kwalifikacji	1	1	1	-0.97	-0.9	0.99	-1	-1	-1
stopień podporządkowania rygorom organizacji	1	1	1	1	1	1	-1	-1	-1

stopień osiągnięcia celu, poziom i jakość w realizacji zadań	1	1	1	-1	-0.7	1	-1	-1	-1
wiek poniżej 30 lat	1	1	1	-1	-1	-1	-1	-1	-1
wiek pomiędzy 30 a 35 lat	-1	-1	-1	-1	-1	-1	-1	-1	-1
wiek pomiędzy 35 a 40 lat	-1	-1	-1	-1	-1	-1	-1	-1	-1
wiek pomiędzy 40 a 45 lat	-1	-1	-1	-1	-1	-1	-1	-1	-1
wiek powyżej 45 lat	-1	-1	-1	1	1	1	1	1	1
wiek odpowiedni	1	1	1	-0.4	-0.4	-0.4	-0.4	-0.4	-0.4
skuteczność działania	0.92	0.92	0.92	-0.91	-0.66	0.89	-0.91	-0.91	-0.91
stan wolny	1	1	1	-1	-1	-1	-1	-1	-1
bezdzietny/a	1	1	1	-1	-1	-1	-1	-1	-1
bez obowiązków alimentacyjnych	1	1	1	-1	-1	-1	-1	-1	-1
żonaty/zamężna	-1	-1	-1	1	1	1	-1	-1	-1
1 dziecko	-1	-1	-1	1	1	1	-1	-1	-1
2 dzieci	-1	-1	-1	-1	-1	-1	-1	- 1	-1
3 lub więcej dzieci	-1	-1	-1	-1	-1	-1	1	1	1
stan rodzinny właściwy	1	1	1	-0.1	-0.1	-0.1	-0.4	-0.4	-0.4
dyscyplina pracy	0.91	0.91	0.91	-0.47	-0.29	0.84	-0.88	-0.88	-0.88
kompetencje techniczne	***0.99***	***0.99***	***0.99***	***-0.99***	***-0.96***	***0.99***	***-0.99***	***-0.99***	***-0.99***
inteligencja	1	1	1	0.2	0.2	0.2	-1	-1	-1
umiejętność podejmowania decyzji	1	1	1	-1	-0.9	-0.8	-1	-1	-1
odporność na stres	1	1	1	-0.5	-0.5	-0.5	-1	-1	-1
entuzjazm	1	1	1	0.7	0.7	0.7	-1	-1	-1
ekspansywność	1	1	1	0	0	0	-1	-1	-1
dobre oceny z przedmiotów ścisłych	1	1	1	-1	-1	-1	-1	-1	-1
umiejętnie ocenia fakty i informacje	1	1	1	-1	-0.8	0.1	-1	-1	-1
zdolności analityczne	0.92	0.92	0.92	-0.92	-0.78	-0.09	-0.92	-0.92	-0.92
racjonalizm	0.64	0.64	0.64	-0.64	-0.55	-0.06	-0.64	-0.64	-0.64
odpowiedzialność	1	1	1	0.2	0.2	0.2	-1	-1	-1
wytrwałość	1	1	1	0.3	0.3	0.3	-1	-1	-1
zdolności negocjacyjne	1	1	1	-1	-0.9	-0.5	-1	-1	-1
elastyczność	0.95	0.95	0.95	-0.95	-0.85	-0.47	-0.95	-0.95	-0.95
umiejętności organizacyjne	1	1	1	-1	-0.5	0	-1	-1	-1
właściwy wygląd zewnętrzny	1	1	1	-1	0.5	1	-1	-1	-1
identyfikacja z firmą	0.2	0.2	0.2	-0.2	0.1	0.2	-0.2	-0.2	-0.2
bogata mowa ciała	1	1	1	1	1	1	-1	-1	-1
podtrzymywanie uwagi	1	1	1	-1	0.8	1	-1	-1	-1
umiejętność współdziałania	1	1	1	-1	0.8	1	-1	-1	-1
inicjatywa	1	1	1	-1	0.7	0.8	-1	-1	-1
komunikatywność	0.89	0.89	0.89	-0.81	0.79	0.88	-0.89	-0.89	-0.89
kompetencje konceptualne	***0.97***	***0.97***	***0.97***	***-0.72***	***-0.58***	***-0.27***	***-0.97***	***-0.97***	***-0.97***
wzbudzanie sympatii	1	1	1	1	1	1	-1	-1	-1
umiejętności kontrolowania	1	1	1	-1	0.9	1	-1	-1	-1
umiejętności motywowania	1	1	1	-1	0.3	0.7	-1	-1	-1
cechy przywódcze	1	1	1	0.9	0.9	0.9	-1	-1	-1
dojrzałość emocjonalna	1	1	1	1	1	1	-1	-1	-1
kompetencje interpersonalne	***0.97***	***0.97***	***0.97***	***-0.05***	***0.93***	***0.96***	***-0.97***	***-0.97***	***-0.97***
WNIOSKI KOŃCOWE									
kierownictwo naczelne	**0.7**			**-0.47**			**-0.7**		
kierownictwo średniego szczebla		**0.7**			**0.03**			**-0.7**	
kierownictwo niższego szczebla			**0.7**			**0.66**			**-0.7**

- **Wersja maksimum** oznacza, iż kandydat otrzymywał możliwie pozytywną ocenę dla wszystkich dopytywalnych cech (dzięki zastosowaniu współczynnika pewności).
- **Wersja pośrednia** to test dla wybranej osoby.
- **Wersja minimum** dotyczy kandydata o możliwie negatywnych cechach.

Dzięki przeprowadzonym testom minimum i maksimum uzyskano potwierdzenie poprawności skonstruowanego systemu ekspertowego.

Współczynniki pewności warunków dopytywalnych (w tabeli 1. znajdują się w formie niewyboldowanej) były deklarowane w trakcie wnioskowania, współczynniki pewności warunków niedopytywalnych zostały wyznaczone na drodze wnioskowania.

Każdemu warunkowi jest przyporządkowany współczynnik pewności, będący liczbą z przedziału [-1,1] i charakteryzujący pewność tego, że warunek jest lub nie jest prawdziwy np.:

CF=1 oznacza warunek, którego prawdziwość jest zupełnie pewna,

CF=0.5 oznacza warunek być może prawdziwy,

CF=0 oznacza warunek o pewności niemożliwej do określenia,

CF=-0.5 oznacza warunek być może nieprawdziwy,

CF=-1 oznacza warunek którego nieprawdziwość jest całkowicie pewna.

Z testów przeprowadzonych dla wybranej osoby wynika, iż jest to kandydat o dużych predyspozycjach do zajmowania stanowiska niższego szczebla w organizacji. Wynik testu (CF=0.66) bliski wartości maksymalnej, jednoznacznie wskazuje, iż istnieje duże prawdopodobieństwo sukcesu podejmując na tej podstawie decyzję o zatrudnieniu.

Kandydata tego cechuje posiadanie bardzo wysokich kompetencji technicznych (CF=0.99), odpowiednich do proponowanego stanowiska. Wynika to ze spostrzeżenia, iż osoba ta ma odpowiednie kwalifikacje oraz doświadczenie (w obu przypadkach CF=1) niezbędne do podjęcia pracy w firmie, ponadto stale podnosi swoje kwalifikacje w tym zakresie (CF=0.99) poprzez odbywane podróże (CF=0.6), konferencje, szkolenia (CF=1), zna ponadto niezbędne zagadnienia z danej dziedziny i potrafi sprawnie wykorzystywać narzędzia istniejące na rynku, które mogą zwiększyć wydajność jej pracy (CF=1). Kandydat, o którym mowa, nie kształcił się za granicą (CF=-1), natomiast posiada liczne zainteresowania, których prawdziwości jednakże nie mogę potwierdzić (CF=0.5). Osoba ta została oceniona jako zdolna do całkowitego podporządkowania się rygorom organizacji (CF=1) i posiadającą potencjał do osiągania maksymalnej jakości w realizacji zadań (CF=1). Biorąc pod uwagę wiek tej osoby (powyżej 45 roku życia), nie jest to kandydat będący w przedziale wiekowym preferowanym przez przedsiębiorstwo (CF(wiek odpowiedni) =0.4), jednak wydaje się, iż pozostałe jego cechy takie jak skuteczność działania (CF=0.89) zdołały podwyższyć ocenę ogólną. Z przeprowadzonego testu na kierownika niższego szczebla wynika ponadto, iż jest to osoba żonata, posiadająca jedno dziecko, system jednakże wskazuje, iż warunek „stan rodzinny właściwy", należy do grupy tych trudnych do określenia (CF bliskie 0).

Kompetencje konceptualne w odróżnieniu od kompetencji technicznych zostały wyznaczone na bardzo niskim poziomie (CF=0.27). Współczynnik pewności określający poziom inteligencji u kandydata nie jest zadawalający (CF=0.2), co więcej osoba ta wpada w zakłopotanie w sytuacji wymagającej podjęcia decyzji (CF=-0.8), prawdopodobnie nie jest ona odporna na stres (CF=-0.5) oraz w niewielkim stopniu potrafi oceniać fakty i informacje (CF=0.1). Ogólnie można stwierdzić, że dany kandydat nie posiada kompetencji konceptualnych, gdyż poszczególne składowe tj. zdolności analityczne (CF=-0.09), racjonalizm (CF=-0.06), odpowiedzialność (CF=0.2), elastyczność (CF=-0.47) nie osiągają w większości wymaganych wartości. Wyjątek w tej grupie stanowi cecha określana mianem komunikatywność (CF=0.88) oraz entuzjazm (CF=0.7).

Trzecią ważną grupą kompetencji wyznaczaną przy pomocy systemu ekspertowego są kompetencje interpersonalne, które bez wątpienia przedstawiają kandydata w bardzo korzystnym świetle (CF=0.96).

Podsumowując, pomimo małego potencjału kompetencji konceptualnych, kandydat ten na pewno sprawdzi się na stanowisku niższego szczebla, o które się ubiega. Intuicyjnie można zauważyć, iż kompetencje konceptualne nie są rodzajem kompetencji, które są niezbędne do wykonywania pracy kierowniczej na najniższym szczeblu. Wynika to z faktu, iż udział procentowy kompetencji konceptualnych w kompetencjach ogółem dla kierownictwa niższego szczebla (patrz tabela 1.) wynosi zaledwie 15% i nie wpływa w sposób istotny na całość oceny. Dużo większe znaczenie mają w tym przypadku kompetencje techniczne (50%) oraz interpersonalne (35%).

Dla tej samej osoby został przeprowadzony test na stanowisko kierownictwa średniego oraz wyższego szczebla. Istotne było znalezienie odpowiedzi na pytanie, czy ów kandydat będzie czerpał satysfakcję z wykonywanej pracy, czy też będzie to posada nie wykorzystująca w pełni jego możliwości, a więc czy istnieje niebezpieczeństwo, iż opuści on firmę w poszukiwaniu awansu i ambitniejszych zadań.

Z przeprowadzonych analiz jednoznacznie wynika, że kandydat ten nie sprawdzi się jako kierownik naczelny (CF=-0.47), poważne wątpliwości istnieją również rozważając decyzję o jego zatrudnieniu jako kierownika średniego szczebla (CF=0.03).

Oceniając na nowo kandydata należało wziąć pod uwagę fakt, iż niektóre cechy są stałe tj. wiek, stopień podporządkowania rygorom organizacji, odporność na stres, entuzjazm, inne natomiast zmieniają się w zależności od szczebla zarządzania.

Na przykład doświadczenie, które jest idealne do zajmowania niższego szczebla w zarządzaniu (CF=1), w żaden sposób nie spełnia wymagań stawianych do zajmowania najwyższej pozycji w organizacji (CF=-1). System ekspertowy w swoisty sposób nagradza cechy, które idealnie przystają do opisu stanowiska. Na przykład udzielając odpowiedzi na warunek dopytywalny odnośnie kwalifikacji osoby, ubiegającej się o najniższe stanowisko, a posiadającej je na bardzo wysokim poziomie, należy udzielić tzw. „nagany" poprzez wpisanie współczynnika pewności CF(odpowiednie kwalifikacje=-1).

Postępowanie takie jest kierowane przekonaniem, iż dobierając osobę na wakujące stanowisko pierwszym i pod-

stawowym warunkiem tegoż jest jasne określenie zadań wobec potencjalnych kandydatów. Precyzyjne opisanie stanowiska i sformułowanie kryteriów sprzyja niezawodności stosowanego systemu ekspertowego, obniża koszty, eliminuje stratę czasu firmy i kandydatów.

4. Podsumowanie

Zaproponowany podział kadry (część A) jak również zaprojektowany system ekspertowy (część B) oraz zastosowane reguły i odpowiadające im współczynniki pewności są przykładem podejścia do problemu doboru kadr. System ten wspomaga podejmowanie decyzji na dużym stopniu ogólności, a więc również może być stosowany do doboru kadry na poszczególne szczeble zarządzania w większości organizacji.

Odpowiedni system ekspertowy pozwala te uwarunkowania lepiej poznać, będąc podstawą do opracowania późniejszych planów rozwoju.

Podobnych zależności jest więcej i warto brać je pod uwagę w procesie tworzenia bazy wiedzy. Zbiór zmiennych, które powinny być uwzględniane podczas konstrukcji systemu ekspertowego, jest niezwykle złożony i nierzadko trudno rozpoznawalny. Wspomniana złożoność wynika nie tylko z liczności czynników, ale także z ich wzajemnego powiązania oraz dynamiki zmian, jakie w nich zachodzą. Konstruując odpowiednie reguły i odpowiadające im współczynniki pewności należy pamiętać, że dobór kadr jest procesem dynamicznym, co oznacza, że powinny one podlegać zmianom (aktualizowaniu), w zależności od rozwoju sytuacji firmy i oddziałujących czynników, a ze względu na specyfikę systemów skorupowych nie powinno to stanowić większej trudności.

Niezbędnym warunkiem jest dysponowanie właściwym systemem informacji w celu prawidłowego dobrania reguł wraz ze współczynnikami pewności oraz precyzyjniejszego określenia współczynników pewności warunków dopytywalnych.

Dobór kadr wymaga więc dużej wyobraźni, a dzięki odpowiednim narzędziom tj. system ekspertowy, postanowienia w tym zakresie mogą być podejmowane z całą starannością, tak aby decyzje kadrowe miały realną użyteczność.

Filozofia doboru kadry, mająca swe odbicie podczas konstrukcji bazy wiedzy, sprowadza się do ustalenia podstawowych wartości. Może się okazać, iż w jakimś przypadku pierwszą, główną kwestią jest posiadanie odpowiednich kwalifikacji, natomiast w innym, priorytetową sprawą jest lojalność przyszłego pracownika (popularna niegdyś nomenklatura).

Należy także wziąć pod uwagę czynniki związane z otwartością organizacji i jej relacjami z podmiotami otoczenia. Spośród nich szczególne znaczenie mają regulacje prawne, rynek pracy, kierunki, poziom i tempo rozwoju gospodarczego kraju, instytucje edukacyjne, związki zawodowe, warunki regionalne.

Korzystanie z właściwie skonstruowanego systemu ekspertowego może przyczynić się do usprawnienia przebiegu procesu doboru kadr, a tym samym do obniżenia kosztów tego przedsięwzięcia, zracjonalizowania procesu kształtowania kadr poprzez dokonywanie trafnych wyborów, właściwego podziału obowiązków, poprawy stopnia wykorzystania kadr.

Najczęściej popełniane błędy podczas konstrukcji bazy wiedzy wykorzystywanej do procesu pozyskiwania kadr to przede wszystkim zbyt ogólnikowe lub zbyt wąskie określenie kryteriów, przemieszanie kryteriów formalnych i merytorycznych, nie tworzących spójnej całości, słabe związanie sformułowanych kryteriów z warunkami dotyczącymi stanowiska (brak właściwych opisów stanowisk), słabe przygotowanie osób do czynności doboru przy zastosowaniu systemu ekspertowego.

Trzeba jednoznacznie stwierdzić, że dobór zasobów ludzkich jest w znacznym stopniu sztuką, dlatego też w celu uniknięcia błędów w tym zakresie niezbędne wydaje się skonstruowanie bazy wiedzy przy pomocy osób blisko związanych z tą problematyką. Może to przybrać formę powołania zespołu zadaniowego, z udziałem doświadczonych menedżerów oraz specjalistów personalnych, bądź – w dużych firmach – utworzenia specjalnej komórki ds. prognozowania i rozwoju kadry. Ważne jest, by stworzony w ten sposób system ekspertowy, uwzględniał zewnętrzne i wewnętrzne uwarunkowania.

Podsumowując, stosowanie systemów ekspertowych jawi się jako interesująca i ważna metoda, pozwalająca na wysoce trafną identyfikację potencjału przyszłej kadry i przewidywanie powodzenia na stanowiskach kierowniczych. Systemy tego typu cechują się dużą obiektywizacją doboru, co w odróżnieniu od na przykład konkursów, będących często fasadą wcześniej podjętych wyborów, stanowi ich dodatkowy atut. Zważywszy, iż ich zastosowanie można rozszerzać na inne pola, takie jak na przykład analiza pracy, wydaje się ona godna zainteresowania.

Warto zauważyć, iż ocena pracowników jest właściwie stałym elementem każdej firmy. Systemy o których mowa pozwalają sformalizować proces oceny osób nie tylko podczas ich przyjmowania do pracy, ale także w trakcie zatrudnienia (umożliwia samodoskonalenie się), pomocne mogą być także przy odejściu, gdy wystawia się im opinię.

Należy sądzić, że wykorzystanie tej metody przez polskie podmioty gospodarcze, mogłoby przyczynić się do poprawy sprawności zarządzania i przynieść wymierne korzyści społeczne i ekonomiczne.

Literatura (References)

[1] Gallant, S.T., „Neural Network Learning and Expert Systems", MIT Press, 1993.

[2] Gaynor G.H., "Exploiting Cycle Time in Technology Management" McGraw-Hill 1993.

[3] Głodek Z., "Integracja systemów wspomagania decyzji i systemów ekspertowych" Zeszyty Naukowe Uniwersytetu Szczecińskiego 137, 1995.

[4] Goldberg D.E., Algorytmy genetyczne i ich zastosowania, WN-T, Warszawa, 1995.
[5] Grensing L., „Jak rekrutować pracowników: odszukaj, wybierz i zatrudnij właściwych ludzi. Przewodnik dla małych i średnich firm", M and A Communications Polska .....
[6] Jackson Peter, „Introduction to Expert Systems", Harlow, England: Addison Wesley Longman, 1999.
[7] Kisielnicki J., "Informatyczna infrastruktura zarządzania" PWN, W-wa 1993.
[8] Klonowski Z., "Implementacja systemów informatycznych w przedsiębiorstwie", Prace Wrocławskiego Centrum Transferu Technologii, Politechnika Wrocławska 1995.
[9] Koch R., „Strategia. Jak opanować i wprowadzić w życie najskuteczniejszą strategię",..., Kraków 1998.
[10] Kuznik-Bąk M., "Rozwój i integracja komputerowo wspomaganych systemów informacyjnych" Zeszyty Naukowe AE we Wrocławiu 474 , 1996.
[11] Lubińska T., "Narzędzia budowy systemów bazy wiedzy" Zeszyty Naukowe Uniwersytetu Szczecińskiego 124, 1994.
[12] Mulawka J.J., „Systemy ekspertowe", Wydawnictwo Naukowe-Techniczne, Warszawa 1996.
[13] Niederliński A., „Regułowe systemy ekspertowe", Wydawnictwo Pracowni Komputerowej Jacka Skalmierskiego, Gliwice 2000.
[14] Nikolopoulos C., „Expert Systems", Marcel Dekker, New York, 1997.
[15] Szajna J., Adamski M., Kozłowski T., „Turbo-Prolog. Programowanie w języku logiki", WNT, Warszawa 1991.
[16] Zieliński J.S., „Inteligentne systemy w zarządzaniu", Wydawnictwo Naukowe PWN, Warszawa 2000.

Projektowanie interaktywnych kokpitów menedżerskich zorientowanych na użytkownika

User-centered interactive performance dashboard designing

Ziuziański Piotr[1], Furmankiewicz Małgorzata[1]

Treść. Artykuł porusza tematykę interaktywnego kokpitu menedżerskiego wspierającego proces podejmowania decyzji w organizacji. Celem publikacji jest przedstawienie procesu komunikacji wizualnej oraz scharakteryzowanie procesu projektowania zorientowanego na użytkownika w kontekście kokpitów menedżerskich. Autorzy przedstawili syntetyczne ujęcie procesu projektowania i wskazali dobre praktyki budowy kokpitu.

Słowa kluczowe: kokpit menedżerski, projektowanie zorientowane na użytkownika, wizualizacja danych

Abstract. The article discusses the topics of interactive performance dashboards supporting decision-making process in organization. The aim of the publication is to present the process of visual communication and characterize the process of user-centered design in dashboards context. The authors present synthetic approach to the design process and pointed out the best practices of building the dashboard.

Keywords: performance dashboard, user-centered design, data visualization

1. Wstęp

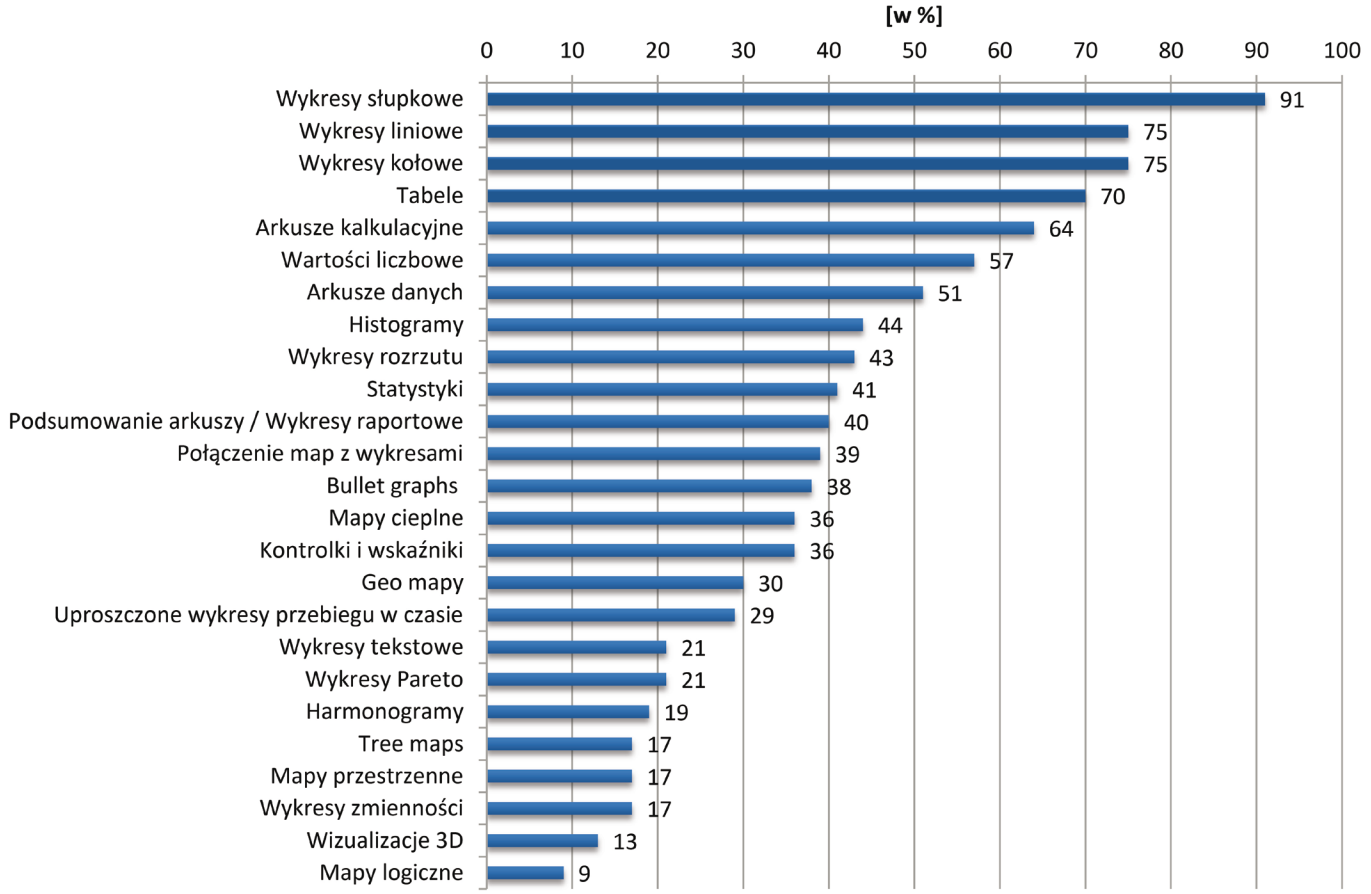
Rzeczywistość organizacyjna wymaga współcześnie podejmowania właściwych decyzji na każdym szczeblu zarządzania w mgnieniu oka. Trafna decyzja to taka, która została podjęta w odpowiednim czasie i daje możliwie najlepsze wyniki w oparciu o dostępne informacje. Zarówno od menedżerów niskiego szczebla jak i wyższego szczebla czy pracowników szeregowych wymaga się podejmowania trafnych decyzji. W takich warunkach powstają narzędzia informatyczne, których zadaniem jest wspieranie procesu podejmowania decyzji. Jednym z przykładów takich narzędzi są kokpity menedżerskie, których projektowanie powinno być możliwie silnie skorelowane z potrzebami użytkownika końcowego. Niniejszy artykuł definiuje istotę kokpitów menedżerskich w kontekście komunikacji wizualnej danych i wiedzy. Ponadto autorzy charakteryzują proces projektowania kokpitów zorientowany na użytkownika.

2. Proces komunikacji wizualnej i wizualna analiza danych

Możliwości percepcyjne człowieka są zdecydowanie większe w przypadku komunikowania treści z wykorzystaniem obrazu (ok. 10^6 bit/s) niż w przypadku czytania odręcznego tekstu (150-300 bit/s) czy słuchania słownej wypowiedzi (ok. 10^4 bit/s) [1]. Ponadto warto zwrócić uwagę na fakt, że człowiek odbiera aż 87% informacji dzięki zmysłowi wzroku (10% dzięki zmysłowi słuchu a 3% dzięki pozostałym zmysłom). Zadaniem menedżerów niejednokrotnie jest analiza danych dotyczących przedsiębiorstwa lub zewnętrznych danych pochodzących spoza niego, wyciągnięcie wniosków i podjęcie decyzji. Ze względu na wskazane możliwości percepcyjne wizualizacja danych, pod pojęciem której kryją się graficzne metody tworzenia, analizy i przekazywania informacji, staje się niezwykle atrakcyjna [2].

Najważniejszym celem wizualizacji danych jest ich prezentacja w czytelny dla odbiorcy sposób, jednocześnie objaśniając korelacje i związki między nimi zachodzące. Jako niewątpliwą zaletę można wskazać możliwość analizy na różnych poziomach szczegółowości. Dobór odpowiedniej formy wizualizacji danych, szczególnie w przypadku rozbudowanych zbiorów danych, czy w przypadku porównywania różnych zbiorów jest zadaniem, które wymaga umiejętności potwierdzonych doświadczeniem. Wizualizacja danych oddziałuje na sposób prowadzenia badań naukowych, regularnie korzystają z niej praktycy biznesu. Wizualizacja wykorzystywana jest także w medycynie, dydaktyce, dyscyplinach technicznych lub nawet jako środek wyrazu artystycznego [2]. W przedsiębiorstwach wykorzystywane są różnorodne formy wizualizacji, co prezentują np. wyniki badań przedstawionych na Ryc. 1.

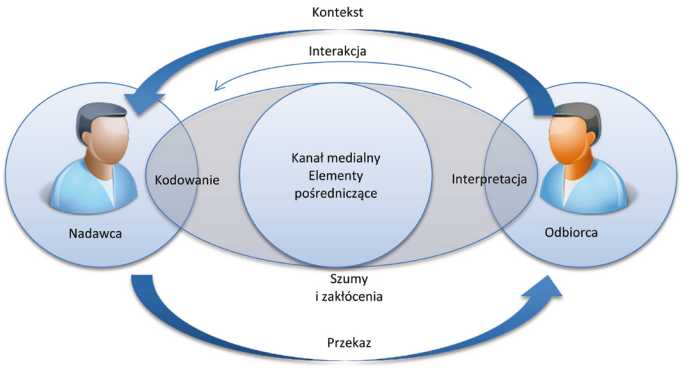
1. Członkowie honorowi Koła Naukowego Scientia Ingenium Uniwersytetu Ekonomicznego w Katowicach, {piotrziuzianski; malgorzata.furmankiewicz}@gmail.com



Ryc. 1. Rodzaje wizualizacji stosowanych w organizacji według raportu TDWI.
Fig. 1. *Types of visualization used in the organization according to TDWI report*

Badanie zostało przeprowadzone przez The Data Warehousing Institute (TDWI) na próbie 352 przedsiębiorstw w 2013r. Oprócz wymienionych form na popularności zyskały także kokpity menedżerskie, które zostaną dokładnie scharakteryzowane w kolejnym rozdziale.

W celu zrozumienia postrzegania i popularności wizualizacji w organizacjach niezbędna jest wiedza na czym polega proces komunikacji. Uproszczony schemat takiego procesu został przedstawiony na Ryc. 2.



Ryc. 2. Schemat procesu komunikacji.
Fig. 2. *Diagram of the communication process.*

Każda transmisja powiązana jest sprzężeniem zwrotnym, które przyjmuje postać interakcji o indywidualnym charakterze [4], np. dla decydentów sprzężeniem zwrotnym dla interpretacji danych może być podjęcie konkretnej decyzji. Komunikacja może przebiegać za pomocą słów w formie pisanej czy mówionej, ale także może odbywać się z wykorzystaniem obrazu. Należy wziąć pod uwagę, że nawet najlepiej przygotowana wizualizacja może powodować pewne szumy (zakłócenia).

Pojęcie „wizualizacja" połączone jest z tzw. postrzeganiem i percepcją. Należy mieć na uwadze to, że są to złożone zagadnienia. Percepcja to proces, który polega na nadawaniu znaczeń otrzymywanym wyróżnieniom. Wpływa na nią wiele czynników, wśród których można wyróżnić:

- sytuację,
- atrybuty (wnioski oparte na obserwacjach zachowania),
- oczekiwania (wobec innych oraz nastrój i potrzeby w danym momencie),
- projekcję (przenoszenie subiektywnych odczuć i systemu wartości na innych),
- percepcję selektywną (filtrowanie bodźców),
- stereotypy.

Każdy przekaz jest odbierany w sposób subiektywny, uzależniony m.in. od wymienionych czynników [4]. Wizualizacja daje możliwość podsumowania i prezentacji nawet bardzo dużych kolekcji danych w przejrzysty sposób, pozwalając tym samym na dostrzeżenie w nich prawidłowości [5]. Zrozumienie możliwości percepcyjnych człowieka, świadomość indywidualnego charakteru postrzegania pracownika czy istnienia szumów komunikacyjnych, powinna być punktem wyjścia dla projektantów, realizatorów i wdrożeniowców rozwiązań wspierających proces podejmowania decyzji, a zwłaszcza rozwiązań służących wizualizacji jak np. kokpit menedżerski.

3. Kokpit menedżerski

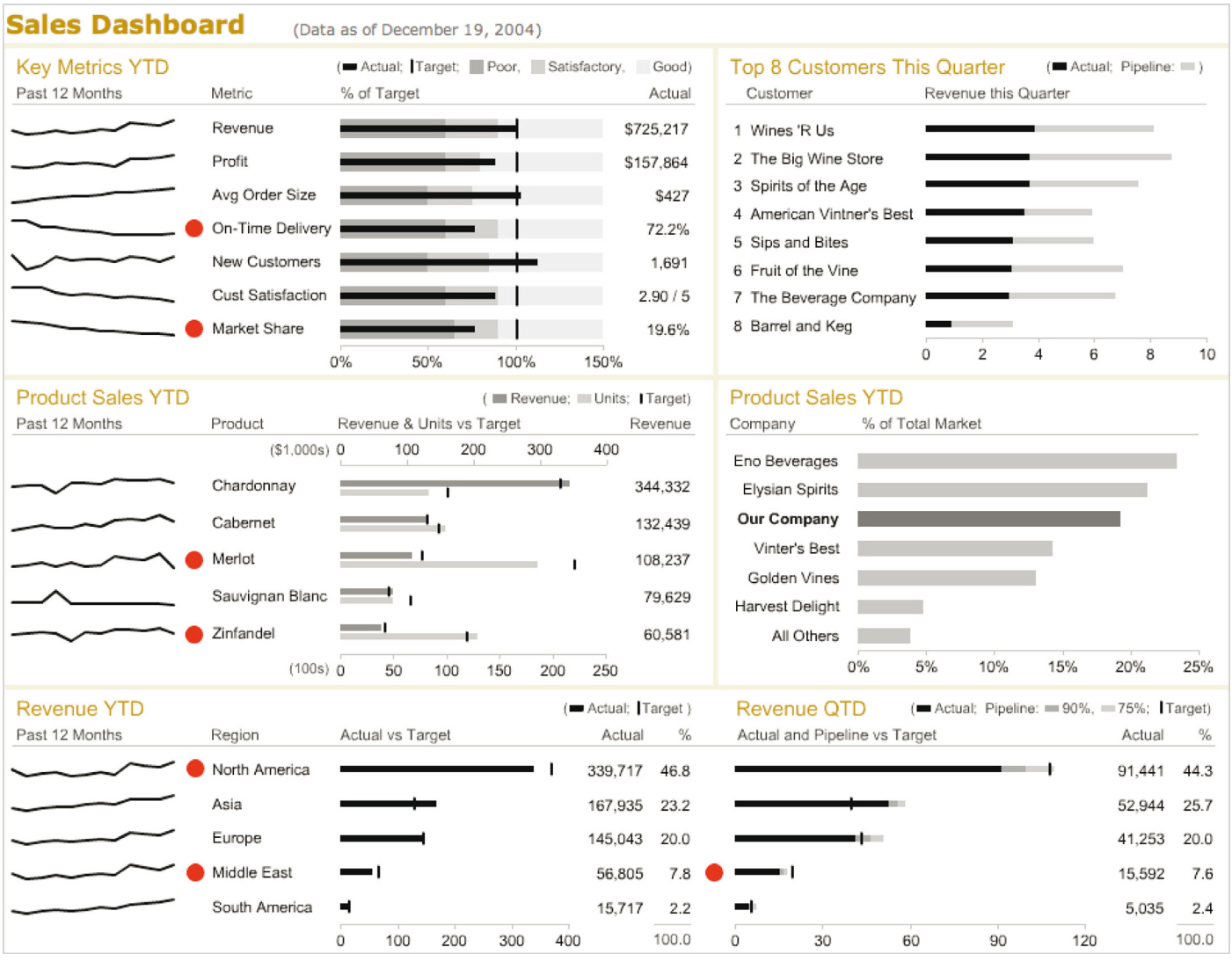
Idea kokpitu opiera się na założeniu, aby dostarczać natychmiastowej informacji menedżerowi o wartościach podstawowych wskaźników oraz sygnalizować niekorzystne zjawiska zachodzące w jego dziedzinie odpowiedzialności [6]. Przykładowe definicje odnalezione w literaturze przedmiotu przedstawiono na Ryc. 3.

Ryc. 3. Wybrane definicje kokpitu menedżerskiego.
Fig. 3. Selected definitions of performance dashboard

Za jedną z lepszych definicji uważa się definicję przedstawioną przez specjalistę z dziedziny wizualizacji danych Stephena Few, który definiuje kokpit jako wizualizację najważniejszych informacji skonsolidowanych na jednym ekranie potrzebnych do osiągnięcia jednego lub wielu celów umożliwiającą szybkie i łatwe monitorowanie: pierwszy rzut oka na kokpit powinien dostarczać użytkownikowi najważniejszych informacji [10] [13].

Charakterystyczne dla kokpitów są graficzne elementy pokazujące poziom danego zjawiska w odniesieniu do oczekiwanych lub niepożądanych wartości, bądź też danych historycznych takie jak: mapy statystyczne, wykresy statystyczne, kontrolki, wskaźniki, sygnalizatory świetlne, liczniki, ikony [6] [7]. Przykładowy kokpit menedżerski prezentuje Ryc. 4.

Zadaniem elementów kokpitu jest dostarczenie ogólnego obrazu procesów przebiegających w organizacji. Jeżeli kokpit sygnalizuje problem, to należy rozpocząć inne, przeważnie bardziej złożone procesy analizy.



Ryc. 4. Przykładowy kokpit menedżerski.
Fig. 4. Example of performance dashboard.

Wśród cech charakterystycznych kokpitów menedżerskich można wskazać [10] [15]:

- wyświetlanie informacji potrzebnych do osiągnięcia konkretnych celów;
- wyświetlanie elementów kokpitu na pojedynczym ekranie komputera;
- śledzenie informacji w krótkim czasie;
- niewielkie, zwięzłe, jasne i intuicyjne mechanizmy prezentacji;
- personalizację.

Natomiast głównym zadaniem kokpitu menedżerskiego jest udostępnianie właściwych informacji właściwym użytkownikom we właściwym czasie w celu optymalizowania procesu podejmowania decyzji, zwiększenia wydajności oraz polepszenia wyników dotyczących działalności organizacji [11].

Można powiedzieć, że kokpit menedżerski jest optymalnym rozwiązaniem łączącym w sobie zalety raportowania i analiz, będąc na średnim poziomie interaktywności [16]. Z drugiej strony należy mieć świadomość, że kokpity bywają całkowicie statyczne jak i dynamiczne, umożliwiając drążenie danych, ich filtrowanie czy agregowanie [10]. Zdefiniowane wymagania w procesie projektowania kokpitu powinno zadecydować o wyborze rodzaju kokpitu.

Kokpit menedżerski wyposażony jest najczęściej w wizualizację tzw. kluczowych wskaźników wydajności (ang. *Key Performance Indicators, KPI*). Ich analiza wspiera zarządzających w procesie podejmowania decyzji [16]. Projektowanie wskaźników KPI to bardzo ważny punkt projektowania kokpitu menedżerskiego. Powinny się charakteryzować: wiarygodnością, aktualnością, przejrzystością i czytelnością. Najważniejsze wskaźniki KPI powinny zostać wyeksponowane na kokpicie menedżerskim [5]. Zaprojektowanie dynamicznego, interaktywnego kokpitu jest zadaniem nieco trudniejszym, gdyż wymaga dokładnego określenia funkcjonalności, które będą przez niego realizowane.

Firmy oferujące rozwiązania Business Intelligence podkreślają spersonalizowany charakter produktów należących do warstwy prezentacji, zapewniając o tym, że rozwiązania te uwzględniają osobiste preferencje użytkownika. W treściach ofert dotyczących kokpitów menedżerskich podkreśla się ścisłe dopasowanie do potrzeb osób z nich korzystających [17] [18] [19] [20]. Przykładowymi kokpitami menedżerskimi są rozwiązania firmy SAP Institute - SAP Business Objects Dashboards [21], IBM - IBM Cognos Business Intelligence [22], Pentaho - Pentaho Business Analytics [23]. Kokpit menedżerski może także zostać wykonany w arkuszu kalkulacyjnym Microsoft Office Excel [24].

4. Projektowanie zorientowane na użytkownika

Projektowanie systemów zorientowanych na użytkownika (ang. *user-centered design, UCSD*) to proces, który skupia się na użyteczności podczas całego procesu wdrożenia jak i w dalszych fazach cyklu życia systemu. Opiera się na kilku następujących zasadach [25]:

- **Skupienie na użytkowniku** – cele działalności, dziedzina pracy lub kontekst użycia, cele użytkowników, zadania powinny towarzyszyć jak najwcześniej projektowi;
- **Aktywny udział użytkownika** – reprezentanci użytkowników powinni aktywnie uczestniczyć od samego początku nieprzerwanie w procesie rozwoju i w całym cyklu życia;
- **Ewolucyjny rozwój systemu** – rozwój systemu powinien być zarówno iteracyjny jak i przyrostowy;
- **Prosta reprezentacja projektu** – projekt musi być w taki sposób zaprezentowany, by mógł zostać zrozumiany w łatwy sposób dla użytkowników i innych zainteresowanych;
- **Prototypowanie** – na początku i nieprzerwanie, prototypy powinny być wykorzystywane do wizualizacji oraz oceny pomysłów i rozwiązań projektowych we współpracy z użytkownikami końcowymi;
- **Ocena użyteczności w kontekście –** wskazane cele użyteczności i kryteria projektowe powinny kontrolować rozwój sytemu;
- **Jawne i świadome działania projektowe –** proces rozwoju powinien zawierać dedykowane działania projektowe;
- **Profesjonalne podejście –** proces projektowania i wdrożenia powinien być wykonywany przez skuteczne zespoły interdyscyplinarne;
- **Najlepsza użyteczność** – eksperci użyteczności powinny być zaangażowani jak najwcześniej i towarzyszyć ciągle w całym cyklu rozwoju systemu;
- **Projektowanie holistyczne** – wszystkie aspekty, które wpływają na przyszłą sytuację stosowanie powinno być rozwijane równolegle;
- **Dostosowanie procesów** – procesy projektowania powinny zostać określone, dostosowane i/lub realizowane lokalnie w konkretnej organizacji;

Syntetycznie ujmując podejście zorientowane na użytkownika, można powiedzieć, że skupia się ono na: odpowiednim podziale funkcji między użytkownikiem a systemem, aktywnym zaangażowaniu użytkowników, iteracyjnych rozwiązaniach projektowych i interdyscyplinarnych zespołach projektowych [26].

Projektowanie kokpitu stanowi pierwszy etap w cyklu życia kokpitu menedżerskiego. Cykl ten, jako podejście iteracyjne do wdrażania kokpitu, został zaprezentowany na Ryc. 5.

Ryc. 5. Etapy cyklu życia kokpitu menedżerskiego.
Fig. 5. Life cycle of performance dashboard.

Faza projektowania skupia się na zaprojektowaniu rozwiązań technicznych i biznesowych, a także wyborze kluczowych wskaźników efektywności. Następnie rozpoczynają się prace nad budową kokpitu, które zawierają w sobie określenie kanału dostępu do danych, wybór narzędzi, opracowanie przypadków testowych. Tak opracowany kokpit podlega testowaniu, a następnie zostaje przeprowadzony pilotaż i po ewentualnych poprawkach kokpit jest eksploatowany. Ostatnia faza cyklu to monitorowanie, a więc ocena informacji zwrotnej i przygotowanie do kolejnej iteracji [27].

Jako że projektowanie kokpitu menedżerskiego powinno być zorientowane na użytkowników to pierwszym krokiem prac projektowych powinna być dokładna identyfikacja odbiorcy/grupy odbiorców i jego/ich specyfiki.

Tab. 1 prezentuje podział użytkowników ze względu na konkretne atrybuty wraz z przykładowymi poziomami.

Tab. 1. Atrybuty użytkowników kokpitu menedżerskiego.
Tab. 1. *Users attributes of performance dashboard.*

Atrybut	Przykładowe poziomy
Doświadczenie	• nowicjusz • ekspert
Wielkości grupy odbiorców	• jedna osoba • wiele użytkowników z takimi samymi wymaganiami lub potrzebujących monitorować podzbiory danych
Stopień decyzyjności	• lider zespołu • prezes
Obszar działalności	• finanse • sprzedaż

Dzięki jawnemu określeniu grupy docelowej można rozpocząć analizę wymagań w oparciu o wywiad. Takie podejście zorientowane na użytkownika pozwala na opracowanie spersonalizowanego kokpitu, który realizuje rzeczywiste potrzeby decydentów. Należy mieć świadomość, że w zależności od szczebla zarządzania można przygotować różne kokpity, które będą różniły się realizowanym celem, udostępnianymi informacjami i aktualizacją danych, co zostało ujęte w Tab. 2.

Tab. 2. Klasyfikacja kokpitów ze względu na szczebel organizacyjny
Tab. 2. *Classification of performance dashboard due to the organizational level.*

Kokpit	Cel	Użytkownicy	Informacje	Aktualizacje
Operacyjny	Monitorowanie operacji	Przełożeni, specjaliści	Szczegółowe	W ciągu dnia
Taktyczny	Mierzenie postępu (analizy)	Menedżerowie, analitycy	Podsumowania i szczegółowe	Codzienne, tygodniowe
Strategiczny	Wykonywanie strategii (zarządzanie)	Kierownictwo	Podsumowania i szczegółowe	Miesięczne, kwartalne

Podstawą projektowania kokpitu powinno być dokładne zdefiniowanie oczekiwań i wymagań docelowego użytkownika kokpitu, powinny one uwzględniać [24]:

- wskaźniki KPI zgodne z założonymi celami;
- przekaz, a więc realizowane przez kokpit cele;
- odbiorców, co pozwala na określenie bardziej szczegółowych wymagań;
- wymiary (grupowanie danych) i filtry (sortowanie danych);
- typ drążenia danych adekwatny do konkretnych miar;
- dostępność i jakość źródeł danych;
- harmonogram aktualizacji danych.

Wybrane KPI w procesie projektowania powinny spełniać kilka zasad: [28]

- skupienie na kilku krytycznych zamiast na wielu błahych,
- powinny prowadzić do realizacji strategii,
- można powiązać je ze wszystkimi szczeblami organizacyjnymi,
- zapewnienie poprawności danych dla KPI,
- powinny dawać możliwość kontroli.

Jak wskazano wcześniej, kokpit menedżerski wykorzystuje różnorakie elementy graficzne, może korzystać z map statystycznych i wykresów statystycznych, które przynależą do grafiki statystycznej. Jedną z najważniejszych kwestii w grafice statystycznej jest zdolność właściwego korzystania z różnych form graficznych w zależności od przeznaczenia wykresu i reprezentowanego typu szeregu statystycznego [29]. Wizualizacja danych wymaga od projektanta nie tylko właściwego doboru formy prezentacji, ale także odpowiedniego doboru poziomu precyzji, czy kolorystyki i rozmieszczenia poszczególnych elementów [5]. Efektywny kokpit charakteryzuje się:

- rozsądną liczbą elementów bez wrażenia natłoku informacji u użytkownika,
- przejrzystością,
- estetycznym wyglądem.

Efekt taki uzyskuje się poprzez wykorzystanie uproszczonych wykresów, czytelne formatowanie liczb, poprawne stosowanie etykiet danych i tytułów. [24].

Czytelnym podsumowaniem dotychczasowych rozważań jest poniższy spis dziesięciu reguł projektanta kokpitów, który może pełnić swego rodzaju dekalog [30]:

1. współuczestnictwo przyszłych użytkowników w procesie projektowania,
2. projektowanie iteracyjne,
3. skupienie na danych,
4. wykorzystanie adekwatnych kluczowych wskaźników efektywności,
5. dopasowanie do indywidualnych potrzeb użytkowników,
6. wykorzystanie zasad projektowania z agencji infor-

macyjnych,
7. zachowanie aktualności danych,
8. dostarczenie opcji drążenia danych w ramach kokpitów,
9. dostarczenie opcji dokonywania operacji na udostępnianych danych,
10. zachowanie oszczędności formy.

Interaktywne kokpity są zdecydowanie bardziej atrakcyjne od statycznych, ponieważ pozwalają na wygodne dotarcie do niezbędnych danych dla decydentów.

5. Podsumowanie

Kokpity menedżerskie są coraz powszechniej stosowane w organizacjach ze względu na zaletę integracji i wizualizacji danych w syntetycznej i wygodnej formie. Zapotrzebowanie na rozwiązania tego typu wynika bezpośrednio z coraz większej konieczności usprawnienia procesu podejmowania decyzji na różnym szczeblu organizacyjnym i w różnych obszarach działalności.

Projektowanie kokpitu menedżerskiego jest procesem złożonym i wymagającym zaangażowania przyszłych jego użytkowników. Tylko takie podejście umożliwia opracowanie kokpitu najlepiej dopasowanego do potrzeb użytkownika, a więc spersonalizowanego i odpowiadającego potrzebom. Dobrze zaprojektowany kokpit menedżerski powinien udostępniać odpowiednie informacje właściwym użytkownikom w adekwatnym czasie dla zoptymalizowania procesu podejmowania decyzji i zwiększenia wydajności organizacji [11].

Literatura (References)

[1] A. Sołtysik, Rozwój badań nad technikami wizualizacji w komputerowym wspomaganiu kreatywności [w:] Rozwój metod i narzędzi dla potrzeb komputerowych systemów wspomagania kreatywności, pr. zb. pod red. S. Stanka i M. Pańkowskiej, Wydawnictwo Akademii Ekonomicznej, Katowice 2010, s. 133.

[2] S. Osowski, Metody i narzędzia eksploracji danych, Wydawnictwo btc, Legionowo 2013, s. 328.

[3] D. Stodder, TDWI best practices report. Data visualization and discovery for better business decisions, TDWI, Renton 2013, s. 27.

[4] A. Benicewicz-Miazga, Grafika w biznesie. Projektowanie elementów tożsamości wizualnej – logotypy, wizytówki oraz papier firmowy, Helion, Gliwice 2012, s. 22.

[5] J. Guzek, Pulpit menedżerski studenta jako narzędzie wizualizacji jego postępów w procesie e-learning [w:] Rola informatyki w naukach ekonomicznych i społecznych. Innowacje i implikacje interdyscyplinarne, Zeszyt 2/2010, pr. zb. pod red. Zbigniewa E. Zielińskiego, Wydawnictwo Wyższej Szkoły Handlowej, Kielce 2010, s. 39–45.

[6] A. Ptasznik, Wszechnica Popołudniowa: Bazy danych. Hurtownie danych – czyli jak zapewnić dostęp do wiedzy tkwiącej w danych, Warszawska Wyższa Szkoła Informatyki, Warszawa 2010, s. 13–14.

[7] P. Ziuziański, Kokpit menedżerski jako efektywne narzędzie do wizualizacji danych w organizacji [w:] Rola informatyki w naukach ekonomicznych i społecznych. Innowacje i implikacje interdyscyplinarne, Zeszyt 1/2014, pr. zb. pod red. Zbigniewa E. Zielińskiego, Wydawnictwo Wyższej Szkoły Handlowej, Kielce 2014, s. 60–69.

[8] Executive Dashboard Implementation Guide 2010, Healthcare Information and Management Systems Society, 2010.

[9] B. Smok, Kokpit menedżerski a system wczesnego ostrzegania [w:] Business Intelligence w zarządzaniu, pr. zb. pod red. B. Smok, Wydawnictwo Uniwersytetu Ekonomicznego we Wrocławiu, Wrocław 2010, s. 145.

[10] S. Few, Information Dashboard Design. Displaying data for at-a-glance monitoring, Analytics Press, Burlingame 2013.

[11] W.W. Eckerson, Performance Dashboards. Measuring, Monitoring and Managing Your Business, John Wiley & Sons, Hoboken 2006, s. 6–9, 18.

[12] B. Marciniak, Systemy wspomagające decyzje marketingowe w przedsiębiorstwach – aspekty teoretyczne i praktyczne [w:] Studia i prace Kolegium Zarządzania i Finansów. Zeszyt Naukowy 110, pr. zb. pod red. K. Kawerskiej, Szkoła Główna Handlowa, Warszawa 2011, s. 58.

[13] L.T. Strome, Healthcare Analytics for Quality and Performance Improvement, John Wiley&Sons, Hoboken 2013, s. 174.

[14] S. Few, dostępne pod adresem: http://www.perceptualedge.com/articles/Whitepapers/Formatting_and_Layout_Matter.pdf

[15] http://skuteczneraporty.pl/blog/tag/dashboard-kokpit-menedzerski/page/2/

[16] A. Sołtysik, Hurtownie danych i narzędzia OLAP w procesach wspomagania decyzji [w:] Inteligentne systemy wspomagania decyzji, pr. zb. pod red. H. Sroki i W. Wolnego, Wydawnictwo Akademii Ekonomicznej im. Karola Adamieckiego w Katowicach, Katowice 2009, s. 198–228.

[17] Extending business Intelligence with dashboards, dostępne pod adresem: ftp://ftp.software.ibm.com/software/au/201011/wp_extending_business_intelligence_with_dashboards.pdf

[18] http://www.sap.com/poland/pc/analytics/businessitelligence/software/dashboards/index.html

[19] Introducing the Discoverer "Drake" Release: Pesnalized Dashboards Supporting OLAP and Relational Access, dostępne pod adresem: http://www.oracle.com/technetwork/developer-tools/discoverer/overview/dashboards-using-oraclebi-discovere-129352.pdf

[20] http://www.microstrategy.com/us/platforms/analytic/business–intelligence

[21] http://www.sap.com/poland/pc/analytics/businessitelligence/software/dashboards/index.html

[22] http://www-03.ibm.com/software/products/pl/business-intelligence

[23] http://www.pentaho.com/product/business-visualiztion-analytics#dashboards

[24] M. Alexander M., J. Walkenbach, Analiza i prezentacja danych w Microsoft Excel, Helion, Gliwice 2011, s. 29-34.
[25] J. Gulliksen, B. Göransson, I. Boivie, S. Blomkvist, J. Persson, A. Cajander, Key principles for user-centred systems design [w:] Behaviour & Information Technology, November–December 2003, vol. 22, nr. 6, Taylor & Francis, 397–409.
[26] J. Gulliksen, A. Lantz, I. Boivie, User Centered Design in Practice - Problems and Possibilities, Technical report TRITA-NA-D9813, CID, January 1999, s. 7.
[27] Executive Dashboard Implementation Guide 2010, Healthcare Information and Management Systems Society, 2010.
[28] http://www.industryweek.com/lean-six-sigma/five-rules-selecting-best-kpis-drive-operational-improvement
[29] K. Kocimowski, J. Kwiatek, Wykresy i mapy statystyczne, Główny Urząd Statystyczny, Warszawa 1976, s. 16.
[30] R. Sherman, Building effective dashboards and scorecards, dostępne pod adresem: http://viewer.media.bitpipe.com/1178304416_32/1255101187_197/Tableau_sDataMgt_SO-23765-EBook_10.5.pdf

PL-Grid - polska infrastruktura przetwarzania danych

PL-Grid – polish data computing infrastructure

Monika Kwiatkowska[1], Łukasz Świerczewski[2]

Treść. Praca prezentuje polską infrastrukturę przetwarzania danych naukowych PL-Grid. W pracy zaprezentowano zarówno teoretyczny zarys możliwości, jak i praktyczne pomiary empiryczne, które wykonano na platformie. Podczas testów wykorzystano algorytm wyszukiwania nieparzystych liczb dziwnych oraz program poszukujący kontrprzykładu dla hipotezy Brocard'a. Wykonano także proste symulacje algorytmu kwantowego Shora z wykorzystaniem udostępnionych akceleratorów graficznych wspierających technologię CUDA.

Słowa kluczowe: przetwarzanie danych, grid, PL-Grid, wirtualne laboratorium.

Abstract. This paper presents polish scientific data computing infrastructure PL-Grid. Paper shows both theoretical outline of the possibilities and practical empiric measurements, which were made via platform. During test there were used: odd weird numbers search algorithm and Brocards conjecture counterexample search program. There were also taken some simple simulations of Shor's quantum algorithm using shared graphics accelerators supporting CUDA.

Key words: computing, grid, PL-Grid, virtual laboratory.

1. Wstęp

Prace nad utworzeniem polskiej infrastruktury PL-Grid rozpoczęto w 2009 roku. W tym roku Akademickie Centrum Komputerowe CYFRONET działające przy AGH w Krakowie zaproponowało innym centrom danych utworzenie krajowego, jednolitego systemu, który miałby służyć polskiej społeczności naukowej.

Infrastrukturą PL-Grid zarządza Konsorcjum PL-Grid, które składa się z następujących instytucji:

- ACK CYFRONET AGH – Akademickie Centrum Komputerowe CYFRONET AGH w Krakowie,
- ICM UW – Interdyscyplinarne Centrum Modelowania Matematycznego i Komputerowego w Warszawie,
- PCSS – Instytut Chemii Bioorganicznej PAN - Poznańskie Centrum Superkomputerowo Sieciowe w Poznaniu,
- CI TASK – Centrum Informatyczne Trójmiejskiej Akademickiej Sieci Komputerowej w Gdańsku,
- WCSS – Wrocławskie Centrum Sieciowo - Superkomputerowe we Wrocławiu.

Wyżej wymienione instytucje w ramach projektu udostępniają naukowcom swoje zasoby obliczeniowe.
Polska Infrastruktura Gridowa jest elementem EGI (ang. European Grid Initiative) [1]. Celem EGI jest integracja narodowych struktur w jeden system.

2. Zasoby obliczeniowe

PL-Grid udostępnia w sumie ponad 230 TFlopsów mocy obliczeniowej oraz ponad 3,6 PBajtów miejsca na dyskach twardych. Użytkownicy gridu uzyskują dostęp do maszyn z pamięcią vSMP [2], klastrów komputerowych oraz nowoczesnych kart graficznych wspierających obliczenia CUDA [3]. Zasoby obliczeniowe udostępnione przez ACK CYGRONET oraz CI TASK przedstawiono w Tab. 1.
Jak widać w pierwszym centrum danych możemy odnaleźć aż 1306 węzłów obliczeniowych. W CI TASK do dyspozycji użytkowników jest nieco mniej - 226 węzłów obliczeniowych.

1. Uniwersytet Marii Curie-Skłodowskiej w Lublinie, Wydział Matematyki, Fizyki i Informatyki, pl. Marii Curie - Skłodowskiej 5, 20-031 Lublin
2. Państwowa Wyższa Szkołą Informatyki i Przedsiębiorczości, Instytut Automatyki i Robotyki, ul. Akademicka 14, 18-400 Łomża

Tab. 1. Zasoby obliczeniowe udostępnione w ramach projektu PL-Grid przez ACK CYFRONET oraz CI TASK.
Tab. 1. *Computing resources shared under PL-Grid project by ACK CYFRONET and CI TASK.*

Ośrodek	Nazwa systemu	Liczba węzłów	Liczba procesorów w węźle	Liczba rdzeni w procesorze	Liczba rdzeni w węźle	GHz	Wielkość pamięci węzła
ACK CYFRONET AGH	Zeus	256	2	4	8	2,5	16 GB
		256	2	6	12	2,26	16 GB
		342	2	6	12	2,66	24 GB
		234	2	6	12	2,4	24 GB
		6	2	6	12	2,4	192 GB
	Zeus vSMP	64	2	6	12	2,66	96 GB
	Zeus GPGPU	24	2	6	12	2,93	72 GB
		20	2	6	12	2,4	96 GB
	Zeus BigMem	104	4	16	64	2,3	256 GB
CI TASK	Galera PLUS	192	2	6	12	2,27	16 GB
		2	32	6	192	2,93	1000 GB
		32	2	6	12	2,93	72 GB

3. Uczestnictwo i granty

Aby stać się użytkownikiem systemu PL-Grid należy być naukowcem z tytułem co najmniej doktora. Konto mogą uzyskać także np. studenci, którzy zostaną zarejestrowani jako podopieczni przypisanego im pracownika naukowego. Aby wykonywać obliczenia naukowe trzeba mieć aktywny grant obliczeniowy. Po przydzieleniu konta użytkownik automatycznie (bez jakichkolwiek dodatkowych formalności) otrzymuje grant osobisty, który uprawnia do wykorzystania 1000 godzin obliczeniowych i 40 GB pamięci masowej. Grant ten podlega odnowieniu co 6 miesięcy. Jeżeli zasoby udostępnione w ramach grantu osobistego nie są wystarczające można wystąpić o przydzielenie grantu właściwego. Grant ten podlega rozliczeniom co ok. 6 miesięcy oraz po jego zakończeniu. W przypadku zakończenia grantu użytkownik ma obowiązek poinformowania PL-Grid o wynikach badań naukowych oraz publikacjach, które dzięki obliczeniom powstały.

4. Wirtualne laboratoria

PL-Grid umożliwia także dostęp do tzw. gridów dziedzinowych z bioinformatyki lub chemii. Użytkownicy zainteresowani bioinformatyką mogą uzyskać dostęp do usługi dotyczącej analizy danych genetycznych. Pozwala ona na uruchomienie programów służących do analizy danych bioinformatycznych, które zostały uzyskane za pomocą sekwenatora genomowego. W ramach PLGrid ICM udostępnia także usługę umożliwiającą długoterminowo i bezpiecznie przechowywać duże ilości informacji genetycznych. System archiwizacji jest oparty o rozwiązanie rozproszone Platon U4, które zapewnia zarówno wysoką niezawodność, jak i trwałość przechowywanych danych.
Na platformie PL-Grid odnajdziemy także usługę InSilicoLab [4] dzięki, której możliwe jest uruchomienie złożonych eksperymentów obliczeniowych z chemii kwantowej.
Użytkownik może uzyskać także dostęp do wielu innych programów naukowych takich jak: Wolfram Mathematica [5], MATLAB [6], ANSYS Fluent [7] lub nawet Blender [8].

5. Liczby Weirda - klastry komputerowe

Większość zasobów obliczeniowych jakie są udostępniane w ramach PL-Grid obejmuje klastry komputerowe [9]. Programowanie tego typu komputerów jest dużo trudniejsze od pisania programów na maszyny z pamięcią wspólną – wymagana jest znajomość specjalnego środowiska MPI.

W teorii liczb liczba dziwna [11] (ang. weird number) to liczba naturalna, której suma wszystkich dzielników jest większa od jej samej. Dodatkowym warunkiem jest to, że suma wszystkich lub niektórych dzielników nie może być równa tej liczbie. Najmniejszą liczbą dziwną jest 70, ponieważ jej dzielniki to: 1, 2, 5, 7, 10, 14 i 35, a ich suma wynosi 74 i nie ma takiego podzbioru dzielników, których suma jest równa 70. Liczbą weirda nie jest natomiast 12, której dzielniki to 1, 2, 3, 4, 6. Ich suma wynosi 16, jednak 2+4+6 wynosi 12. Do dnia dzisiejszego nie odnaleziono żadnej nieparzystej liczby dziwnej i nie wiadomo czy takie istnieją. Spadek czasu realizacji przeszukiwania zakre-

su 10^{11} liczb w poszukiwaniu nieparzystej liczby dziwnej dla różnej ilości procesów na klastrze komputerowym PL-Grid WCSS Supernova przedstawiono w Tab. 2.

Tab. 2. Spadek czasu realizacji przeszukiwania zakresu 10^{11} liczb w poszukiwaniu nieparzystej liczby dziwnej dla różnej ilości procesów na klastrze komputerowym PL-Grid WCSS Supernova.
Tab. 2. *The decrease of search time of numbers from range of 10^{11} in search of an odd weird number for different numbers of processes on a computer cluster PL-Grid WCSS Supernova.*

Ilość procesów	Czas realizacji obliczeń
1	6241 s
2	3142 s
3	2765 s
4	1674 s
8	967 s
16	521 s
24	411 s
32	282 s
40	252 s
48	223 s
56	198 s
64	176 s
128	89 s

Program ten zrównoleglił się prawie idealnie. Doskonale widać to podczas porównywania wydajności uzyskanej przy 64 i 128 procesach. Dla 64 procesów czas realizacji zadania wyniósł 176 sekund, a dla 128 spadł niemal dwukrotnie – do 89 sekund.

6. Symulacja algorytmu Shora - akceleratory graficzne

Informatyka kwantowa jest dość nową gałęzią informatyki. Niedawno, w 1994 roku Peter Shor zaprezentował kwantowy algorytm faktoryzacji o złożoności wielomianowej. W pracy [12] zaprezentowano możliwość przeprowadzenia symulacji algorytmu Shora na kartach graficznych wspierających standard CUDA oraz OpenCL [13]. Wyniki uzyskane w pracy [12] (sześć pierwszych kart graficznych) oraz pomiary uzyskane na platformie PL-Grid zaprezentowano w Tab. 3.

Tab. 3. Czas wykonywania faktoryzacji liczby 711 za pomocą symulacji algorytmu Shora z wykorzystaniem OpenCL (dla AMD) oraz CUDA (dla nVidii). Źródło: [12] rozwinięte o wyniki własne uzyskane na platformie PL-Grid.
Tab. 3. *Time of factoring number 711 with help of Shor's algorithm simulation, using OpenCL* (*for AMD*) *and CUDA* (for *nVidia*). *Source*: [12] *expanded by own results gained on the PL-Grid platform.*

	Czas minimalny	Czas średni	Czas maksymalny
nVidia Tesla C2050	7,12 s	24,568 s	832,80 s
nVidia Tasla C1060	8,08 s	26,342 s	976,50 s
nVidia GeForce GTS250	12,67 s	58,786 s	1864,32 s
nVidia GeForce 9800 GT	58,09 s	276,895 s	2654,44 s
nVidia GeForce 9800 GX2	60,84 s	290,438 s	2743,01 s
AMD Radeon 7770	5,54 s	25,875 s	743,76 s
PLGrid CYFRONET Zeus GPGPU nVidia Tesla M2050	7,09 s	22,785 s	825,97 s
PLGrid CYFRONET Zeus GPGPU nVidia Tesla M2090	7,11 s	22,568 s	876,82 s
PLGrid ICM UW Hydra nVidia GeForce GTX 480	8,06 s	29,765 s	1007,98 s
PLGrid ICM UW Hydra nVidia Tesla K20	5,26 s	20,502 s	792,43 s

Najnowocześniejszą kartą graficzną w zestawieniu jest nVidia Tesla K20 dostępna w ICM UW i to ona uzyskała zdecydowanie najlepsze wyniki czasowe. Nieco gorzej (ale i tak najlepiej zaraz za K20) wypadły akceleratory graficzne nVidia Tesla M2050 oraz nVidia Tesla M2090 zainstalowane w klastrze komputerowym Zeus GPGPU. Nieco słabszą kartą graficzną jest GeForce GTX 480 – okazał się niewiele wolniejszy od nVidia Tesla C1060.

7. Hipoteza Brocard'a

W teorii liczb hipoteza Brocard'a mówi, że zawsze można znaleźć co najmniej cztery liczby pierwsze pomiędzy $(p_n)^2$, a $(p_{n+1})^2$ dla $n > 1$ gdzie p_n określa n-tą liczbę pierwszą. Przykładowe dane prezentujące hipotezę dla pierwszych wartości n przedstawiono w Tab. 4.

Tab. 4. Przykładowe dane prezentujące hipotezę Brocard'a.
Tab. 4. *Example data presenting Brocards conjecture.*

n	p_n	$(p_n)^2$	P_{n+1}	$(p_{n+1})^2$	Liczby pierwsze	Ilość liczb pierwszych
1	2	4	3	9	5, 7	2
2	3	9	5	25	11, 13, 17, 19, 23	5
3	5	25	7	49	29, 31, 37, 41, 43, 47	6
4	7	49	11	121	53, 59, 61, 67, 71	5

Napisane oprogramowanie do obliczeń wykorzystuje listę liczb pierwszych. Lista ta na początku jest generowana za pomocą sita Eratostenesa. Krok ten jest najbardziej wymagającym obliczeniowo etapem tego programu. Także głównym ograniczeniem jest sama lista liczb pierwszych, ponieważ program dla dużych parametrów wejściowych potrzebuje bardzo dużo pamięci operacyjnej. Szczegółowe porównanie systemu PL-Grid CI TASK Galera PLUS vSMP z komputerem UMCS opartym o procesor Intel Core i7 950 i wykorzystującym 24 GB pamięci RAM podczas analizy hipotezy Brocard'a zaprezentowano w Tab. 5.

Tab. 5. Wyniki czasowe jakie uzyskano na systemie PL-Grid CI TASK Galera PLUS oraz komputerze UMCS z procesorem Intel Core

i7 950 podczas analizy hipotezy Brocard'a.
Tab. 5. *The results obtained at the PL-Grid CI TASK Galera PLUS system and UMCS computer with Intel Core i7 950 while analyzing Brocards hypothesis.*

Rozmiar wykorzystanej pamięci	PLGrid CI TASK Galera PLUS vSMP	UMCS Intel Core i7 950 24 GB RAM	Rozmiar sita Eratostenesa	Ostatnia liczba n, dla której sprawdzono hipotezę
1 GB	119 s	121 s	8589934592	8951
5 GB	665 s	670 s	42949672960	18566
7 GB	948 s	958 s	60129542144	21551
15 GB	1244 s	2166 s	128849018880	30680
22 GB	3251 s	3294 s	188978561024	36536
50 GB	11999 s	-	429496729600	53227
75 GB	> 259200 s	-	644245094400	-
100 GB	> 259200 s	-	858993459200	-

Jak widać pomimo nieznacznie niższej częstotliwości taktowania (2,93 GHz) procesor zainstalowany w systemie vSMP klastra komputerowego Galera PLUS udostępnionego w ramach PL-Grid okazał się szybszy od komputera opartego na procesorze Core i7 950 (3.07 GHz). Jednak pomimo posiadania teoretycznie aż 1000 GB pamięci (dla użytkownika dostępne ok. 920 GB) okazało się, że operowanie na tak dużych ilościach danych nie jest możliwe. Wygenerowanie sita Eratostenesa już dla 75 GB wykraczało poza czas kolejki, który wynosi 72 godziny. W czasie 72 godzin każde zadanie na systemie vSMP powinno się zakończyć – w przeciwnym wypadku zostanie automatycznie zabite przez platformę. Kod odpowiedzialny za sito Eratostenesa zaprezentowano na Listingu 1.

Listing. 1. Możliwa implementacja sita Eratostenesa z wykorzystaniem operacji bitowych.
Listing. 1. *Possible implementation of the sieve of Eratosthenes using bit operations*
void eratosthenes_sieve_bit(char numbers[], unsigned long long int limit)

```
{
   unsigned long long int i, j;

   clear_bit(numbers[0], 0);
   clear_bit(numbers[0], 1);

   for (i=2; i<limit; i++)
   {
      set_bit(numbers[i/8], i%8);
   }

   for (j=4; j<limit; j+=2)
   {
      clear_bit(numbers[j/8], j%8);
   }

   for (i=3; i<sqrt(limit); i+=2)
   {
      if ( check_bit(numbers[i/8], i%8) )
      {
         for (j=2*i; j<limit; j+=i)
         {
            clear_bit(numbers[j/8], j%8);
         }
      }
   }
}
```

W kodzie wykorzystano makra clear_bit(), set_bit() oraz check_bit() umożliwiające wykonywanie operacji na bitach. Najprawdopodobniej system vSMP źle sobie radzi z tego typu sekwencyjnym odczytem/zapisem do pamięci i dlatego wynikł tak długi okres realizacji zadań. Należy pamiętać, że platforma vSMP to głównie oprogramowanie wirtualizacyjne, które umożliwia utworzenie jednego komputera z dużą wirtualną pamięcią, z dziesiątek zwykłych serwerów. Dane, które nie są dostępne w ramach danego węzła obliczeniowego muszą być pobrane za pośrednictwem znacznie wolniejszej sieci z innej maszyny.

8. Wnioski

PL-Grid stanowi bardzo ciekawą propozycję dla współczesnego naukowca, który często do pracy może wykorzystywać komputery o dużych mocach obliczeniowych. Praca prezentuje aspekt wykorzystania tej platformy i zarówno pod względem symulacji algorytmu kwantowego Shora, jak i poszukiwania nieparzystej liczby dziwnej uzyskano interesujące rezultaty czasowe. Pod znakiem zapytania można jedynie pozostawić wykorzystanie systemu vSMP do rozwiązywania hipotezy Brocard'a. Należy jednak dodać, że Infrastruktura PL-Grid umożliwia wykorzystanie dwóch systemów vSMP – zainstalowanego w klastrze CI TASK Galera PLUS oraz ACK CYFRONET Zeus. Przetestowano jedynie możliwości pierwszego systemu z całkowitym pominięciem możliwości systemu Zeus, który być może byłby szybszy. Ciekawe podejście dotyczące przemysłu rolno-spożywczego (centra informacji) chociaż z trochę odmiennego punktu widzenia zaprezentowano w [14].

Podziękowania

Praca została wykonana z wykorzystaniem Infrastruktury PL-Grid.
Praca została przygotowana z wykorzystaniem zasobów superkomputerowych udostępnionych przez Wydział Matematyki, Fizyki i Informatyki Uniwersytetu Marii Curie-Skłodowskiej w Lublinie.

Literatura (References)

[1] D. Kranzlmüller, J. Marco de Lucas i P. Öster. "The European Grid Initiative (EGI)." *Remote Instrumentation and Virtual Laboratories*. Springer US, 2010, s. 61-66.
[2] Schmidl, Dirk, et al. "How to scale nested openmp applications on the scalemp VSMP architecture." *Cluster*

Computing (CLUSTER), 2010 IEEE International Conference on. IEEE, 2010.
[3] J. Nickolls, et al. "Scalable parallel programming with CUDA." *Queue* 6.2 (2008): 40-53.
[4] J. Kocot, et al. "InSilicoLab–Managing Complexity of Chemistry Computations." *Building a National Distributed e-Infrastructure–PL-Grid*. Springer Berlin Heidelberg 2012, s. 265-275.
[5] S. Wolfram, *Mathematica: a system for doing mathematics by computers*. Addison-Wesley, Reading, 1988.
[6] M. S. Grewal i Angus P. Andrews. *Kalman filtering: theory and practice using MATLAB*. Wiley. com, 2011.
[7] A. Fluent, "12.0 User's Guide." *User Inputs for Porous Media* (2009): 6.
[8] R. Hess, *The essential Blender: guide to 3D creation with the open source suite Blender*. No Starch Press, 2007.
[9] R. Buyya, "High Performance Cluster Computing: Architectures and Systems (Volume 1)." *Prentice Hall, Upper SaddleRiver, NJ, USA* 1 (1999), s. 999.
[10] S. J. Benkoski i P. Erdős, "On weird and pseudoperfect numbers." *mathematics of computation* 28.126 (1974), s. 617-623.
[11] Vandersypen, Lieven MK, et al. "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance." *Nature* 414.6866 (2001): 883-887.
[12] Ł. Świerczewski, Symulacja funkcjonalnego systemu kwantowego na równoległych komputerach klasycznych IV generacji. (2013).
[13] J. E. Stone, D. Gohara i Guochun Shi. „OpenCL: A parallel programming standard for heterogeneous computing systems." *Computing in science & engineering* 12.3 (2010), s. 66.
[14] H. Zarzycki, E. Fronczak, *A Practical Approach To Computer System Architecture For An Agro-Food Industry Information Center*. PSZW, nr 33, s. 248-255, Bydgoszcz 2010.

System rozpoznawania mowy z ograniczonym słownikiem

Speech recognition system with limited dictionary

Dawid Grabowski[1], Monika Kwiatkowska[2] i Łukasz Świerczewski[1]

Treść. Motywacją w pisanej pracy jest omówienie i porównanie popularnych algorytmów rozpoznawania mowy na różnych systemach. Zebrane informacje są przedstawione w stosunkowo krótkiej formie, bez wnikliwej analizy dowodów matematycznych, do których przedstawienia i tak potrzebne jest odniesienie się do odrębnych specjalistycznych źródeł. Omówione zostały tutaj problemy pewne związane z ASR (ang. Automatic Speech Recognition) i perspektywy na rozwiązanie ich. Na podstawie dostępnych rozwiązań stworzony został moduł aplikacji umożliwiający porównywanie zebranych nagrań pod kątem podobieństwa sygnału mowy i przedstawienie wyników w formie tabelarycznej. Stworzona biblioteka w celach prezentacyjnych została użyta do pełnej aplikacji umożliwiającej wykonywanie rozkazów na podstawie słów wypowiadanych do mikrofonu. Wyniki posłużą nie tyle za ostateczne wnioski w tematyce rozpoznawania mowy, co za wskazówki do kolejnych analiz i badań. Mimo postępów w badaniach nad ASR, nadal nie ma algorytmów o skuteczności przekraczającej 95%. Motywacją do dalszych działań może być np. społeczne wykluczenie ludzi nie mogących posługiwać się komunikacją polegającą na wzroku.

Słowa kluczowe: Rozpoznawanie mowy, ASR, MFCC.

Abstract. Motivation of this thesis is discussion about popular ASR algorithms and comparision on various architectures. Collected results are presented in relatively short shape. It's done without math argumentation because it could depend on complicated equations. Here are discussed some problems associated with ASR (Automatic Speech Recognition) and the prospects for a solution to their. On the basis of available solutions it was developed application module that allows comparison of collected recordings in respect of similarity of the speech signal and present the results in tabular form. For presentation purposes it has been created a library and it was used in complete application that allows execution of commands based on the words spoken to microphone. The results will be used not only for the final conclusions about ASR, what clues for further analysis and research. Despite the advances in research on ASR, still there are no algorithms for effectiveness in excess of 95%. The motivation for further actions may be, eg, the social exclusion of people who can not use the communication involving the eye.

Keywords: speech recognition, ASR, MFCC.

1. Wstęp

Celem rozwoju technologii rozpoznawania mowy jest umożliwienie człowiekowi komunikacji z maszyną przy pomocy mowy. W ostatnich latach nastąpił postęp rozwoju dziedziny przetwarzania mowy, między innymi za sprawą dostępnych zasobów w postaci słowników. To spowodowało, że ASR (ang Automatic Speech Recognition) stało się jedną z ważniejszych dziedzin DSP (ang. Digital Sound Processing). Mowa posiada pewne charakterystyczne cechy. Jedną z nich było zawieranie się częstotliwości sygnału mowy w pewnym spektrum, ograniczonym w stosunku do słyszalnego. Pozwoliło to na opracowanie algorytmów kompresji danych specyficznych dla konkretnych dźwięków.

Okazuje się, że optymistycznie zapowiadający się postęp cyfryzacji zaczął ograniczać np. niewidomych. Rozpowszechnienie komputerów i innych wszelkiego rodzaju urządzeń elektronicznych a także wzrost ich mocy obliczeniowej [9, 12] otworzyły nowe perspektywy dla systemów rozpoznawania mowy, które pomagają np. osobom, które mają bardzo utrudnione wprowadzanie danych do systemu.

Choć niektóre algorytmy, np. ukryte modele Markowa pozostały niemal bez zmian, stanowią bazę dla innych, wykorzystujących bardziej wyrafinowane metody postępowania. Bardzo ważnym elementem projektowania systemów automatycznego rozpoznawania mowy jest

1. Państwowa Wyższa Szkoła Informatyki i Przedsiębiorczości, Instytut Automatyki i Robotyki, ul. Akademicka 14, 18-400 Łomża
2. Uniwersytet Marii Curie-Skłodowskiej w Lublinie, Wydział Matematyki, Fizyki i Informatyki, pl. Marii Curie-Skłodowskiej 5, 20-031 Lublin

złożoność obliczeniowa. Projektowanie algorytmu stawia wybór pomiędzy dokładnością a szybkością. Sieci neuronowe i technologie typu GPGPU (*ang General-Purpose computing on Graphics Processing Units*) [8], pozwalające programować szybkie procesory graficzne, wydają się przybliżać możliwość posiadania dokładności i szybkości w jednej aplikacji.

Głos różny dla każdego człowieka, może być jednym z elementów systemów zabezpieczeń. Ponieważ te systemy polegają na zezwalaniu dostępu co najwyżej kilku osobom, nie ma potrzeby weryfikacji danych wejściowych pod względem innych osób. To znacząco obniża koszt czasowy i pamięciowy. Podobnymi, prostszymi mechanizmami dysponuje większość telefonów komórkowych posiadających opcję wybierania głosowego [11].

Powyższe rozwiązania nie są pozbawione wad. Sekwencje słów wprowadzone jako klucze w systemach zabezpieczeń mogą być ciężkie do zapamiętania i do powtórzenia. Do tego należy wziąć pod uwagę nastrój osoby w danym momencie, albo choroby związane z układem artykulacji (np. chrypka). Z jednej strony algorytmy rozpoznawania mowy są projektowane tak, aby radzić sobie z tego typu rozbieżnościami, z drugiej jednak architekci systemów zabezpieczeń nie mogą pozwolić sobie na zbytnią dowolność, gdyż to umożliwiało by włamania.

1.1. Cel pracy

Motywacją w pisanej pracy jest omówienie ważniejszych aspektów z dziedziny automatycznego rozpoznawania mowy i przedstawienie popularnych algorytmów. Zebrane informacje są przedstawione w stosunkowo krótkiej formie, bez wnikliwej analizy dowodów matematycznych, do których przedstawienia i tak potrzebne jest odniesienie się do odrębnych specjalistycznych źródeł. Omówione zostały tutaj problemy pewne związane z ASR i perspektywy na rozwiązanie ich.

Na podstawie rozpatrywanych rozwiązań i algorytmów zostanie zrealizowany system automatycznego rozpoznawania mowy z ograniczonym słownikiem oraz aplikacja graficzna wykorzystująca mechanizmy rozpoznawania mowy do sterowania. Następnie zostanie dokonany krótki test i zebranie wyników w formie tabelarycznej.

2. Charakterystyka języka mówionego

Nie ulega wątpliwości, iż mowa jest czymś bardziej naturalnym dla człowieka, niż dla maszyny. Dlatego aby umożliwić interpretację głosu z zadowalającym efektem przez maszynę, należy zrozumieć działanie narządu głosu (mowy i słychu) człowieka, następnie wiedzę tę należy usystematyzować. W tym rozdziale przedstawiono podstawowe informacje niezbędne podczas tworzenia interfejsu umożliwiającego komunikację człowieka z komputerem.

2.1. Dźwięk

Dźwięk jest skutkiem drgań powietrza o charakterze falowym. Innymi słowy to seria określonych zmian ciśnienia powietrza [1, 15]. Charakteryzuje je kilka cech:

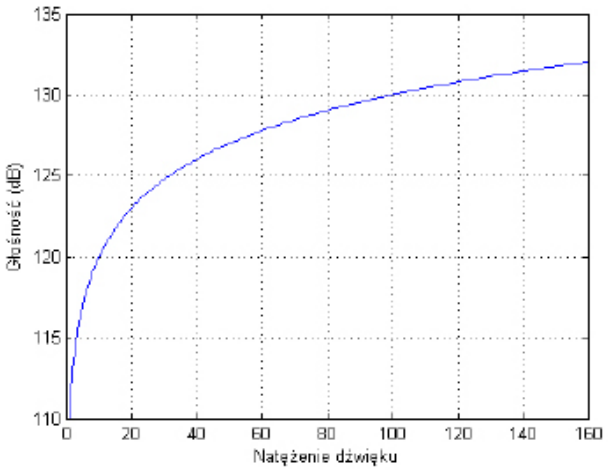
- amplituda – wychylenie z położenia równowagi – ciśnienie akustyczne,
- długość fali – czas trwania pojedynczego wychylenia,
- częstotliwość – ilość drgań w ciągu jednej sekundy.

Długość fali jest mierzona w ułamkach sekund i odbierana niezależnie od obserwatora. Pozostałe cechy mają pewne różnice w pomiarze i odbiorze przez człowieka. Zarówno amplituda jak i częstotliwość mierzone są w skali liniowej, w przeciwieństwie do subiektywnej skali odbiorcy. Ciśnienie akustyczne odczuwalne jest jako głośność i wyraża się ja w belach (decybelach). Nie każda zmiana amplitudy powoduje wyczuwalną zmianę głośności. Odbywa się to w skali logarytmicznej zgodnie z prawem Webera-Fechnera – różnice przy małej głośności są wyraźnie wyczuwalne, a różnice przy dużej głośności są praktycznie nierozróżnialne [1, 15]. Zależność tę wyraża się wzorem:

$$dB_x = 10\log_{10}\left(\frac{x}{10^{-12}}\right) W/m^2 \tag{1}$$

gdzie: *x – częstotliwość w Hz, W – moc wyrażona w Watt, m - metr*

Reprezentuje ją Ryc 1.



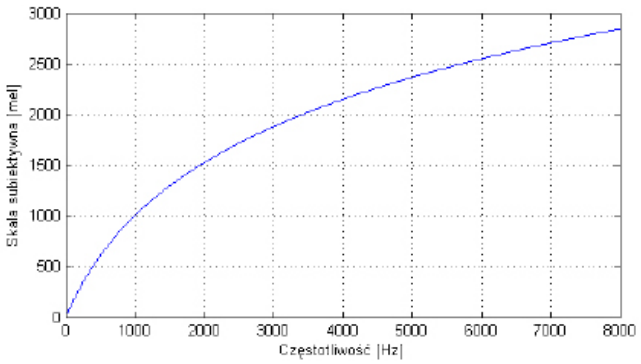
Ryc. 1 Zależność głośności w skali dB między natężeniem dźwięku.
Fig. 1. *The dependence between the volume in dB scale and the intensity of a sound.*

Częstotliwość odbierana przez ludzkie ucho wyrażana jest w skali mel (*ang mel- scale*). Polega ona na zależności między częstotliwością czystego tonu harmonicznego i częstotliwością postrzeganą przez człowieka. Jest ona wyrażana w skali mel. Zależność między Hz i mel jest nieliniowa [1,15].

$$f_{mel}(f_{Hz}) = 2595\log_{10}\left(1+\frac{f_{Hz}}{700}\right)$$
$$f_{Hz}(f_{mel}) = 700\left(10^{\left(\frac{f_{mel}}{2595}\right)}-1\right) \tag{2}$$

gdzie: f_{mel} – *częstotliwość w skali mel,* f_{Hz} – *częstotliwość w* Hz

Według wyników doświadczeń, nieliniowe przetwarzanie częstotliwości zwiększa skuteczność systemów automatycznego rozpoznawania mowy. Powyższy związek między obiema skalami reprezentuje Ryc. 2 [1].



Ryc. 2. Zależność skali wyrażonej w dB i skali mel.
Fig. 2. *The dependece in dB scale and mel scale.*

2.2. Narząd mowy i fonetyka

Mowa w ogólnym pojęciu oznacza usystematyzowane dźwięki wydawane przez narząd mowy u człowieka. Powstawanie tych dźwięków polega na modyfikacji przepływu wydychanego powietrza za pomocą ust, języka i nosa. Zestawienie samogłoski i spółgłoski stanowi sylabę i tym samym część słowa.

2.3. Narząd słuchu i percepcja mowy

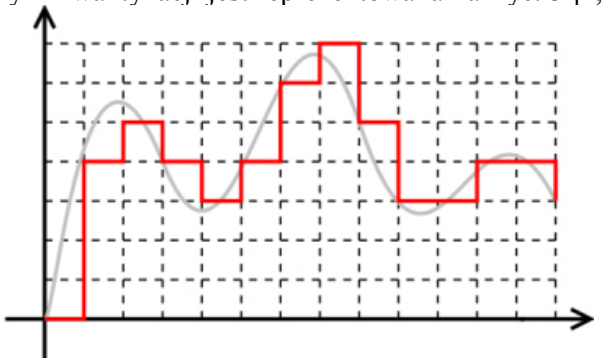
Ucho ludzkie odpowiada za odbieranie dźwięków z otoczenia oraz za utrzymanie równowagi. Przekształca fale dźwiękowe na impulsy nerwowe. Składa się z trzech sekcji: ucha zewnętrznego, środkowego i wewnętrznego. Przekształcenie fal dźwiękowych na sygnał elektryczny dzisiaj nie stanowi większego problemu dla projektujących wszelką elektronikę. Zdecydowanie najpoważniejszym wyzwaniem jest ustalenie kontekstu dźwięku (np. odróżnienie wypowiedzi zależnie od intonacji, dialektyzacja). To, co wydaje się siłą algorytmów implementowanych w programach komputerowych, a mianowicie pewna „odporność" na zmiany dźwięku, staje się też ich wadą. Jeśli przetwarza się słowo jako całość, należy wziąć pod uwagę jego warianty. Dobrym rozwiązaniem wydaje się nadanie sensu dla każdej możliwej wymowy słowa i zapamiętanie. To jednak stwarza kolejne problemy związane z pamięcią [8, 10, 11].
Praktycznym podejściem do problemu jest interpretacja mowy jako ciąg głosek. Mając do dyspozycji difony, czyli przejścia pomiędzy sąsiadującymi głoskami, można próbować dopasowania do kontekstu. Najczęściej wypowiedź traktuje się jako serię difonów, czyli stan przejść w trakcie wymowy tak, jak ma to miejsce w przypadku rozróżniania kontekstu przez człowieka [8, 10, 11].

3. Cyfrowe przetwarzanie sygnałów

Cyfrowe przetwarzanie sygnałów stanowi dziedzinę nauki zajmującą się sygnałami cyfrowymi i operacjami na nich. Ważnymi zastosowaniami DSP (ang. Digital Sound Processing) są: kompresja dźwięku, kodowanie obrazu, telekomunikacja, rozpoznawanie mowy. Pierwszym etapem działania każdego układu DSP jest przetworzenie sygnału analogowego na cyfrowy – część za to odpowiadającą nazywa się przetwornikiem analogowo-cyfrowym. Od lat inżynierzy wspomnianych układów zmagają się z tym samym problemem: im lepszy układ, tym bardziej skomplikowane wykonanie [1, 2, 3, 8, 10, 11].

3.1. Reprezentacja dźwięku w komputerze

Karty dźwiękowe stały się nieodłącznym elementem każdej płyty głównej. Zintegrowany układ SPU (ang. Sound Processing Unit) pozwala na podstawowe operacje: przechwytywanie i odtwarzanie dźwięku z przyzwoitą jakością. Dźwięk może zostać przechwycony różnymi urządzeniami peryferyjnymi, jednak aby móc przechowywać sygnał w komputerze, wszystkie przekształcają go na postać cyfrową. Operacja konwersji sygnału analogowego na cyfrowy wiąże się z nieodwracalną utratą informacji. Próbkowanie danych polega na tym, że z sygnału wejściowego pobierane są próbki w stałym okresie czasu. Im częściej (gęściej) ułożone, mniejsze przerwy w nich występują, tym mniej danych zostało utracone. Jednostką pomiarową dla próbkowania jest herc. Im wyższa jest częstotliwość próbkowania tym wyższa jest maksymalna częstotliwość dźwięku jaką można zapisać. Maksymalna częstotliwość dźwięku jaką można odtworzyć bez zniekształceń z danych na jedynym kanale to 22050 Hz [1, 19].
Kwantyzacja określa skończony zbiór wartości dla pojedynczej próbki (precyzji zmian głośności). Kwantyzacja jest również informacją o maksymalnym możliwym wychyleniu amplitudy. Parametr ten wyraża się w bitach, co ma proste przełożenie na podstawowe typy danych w komputerze, np. 64 dla typu zmiennopozycyjnego podwójnej precyzji. Różnica pomiędzy sygnałem rzeczywistym a poddanym kwantyzacji jest reprezentowana na Ryc. 3 [1, 19]



Ryc. 3 Porównanie wykresu sygnału wejściowego i odpowiadającego mu sygnału dyskretnego.
Fig. 3. *The comparison of the input signal and the corresponding discrete signal.*

Na czerwono zaznaczono sygnał dyskretny. Siatka złożona z przerywanych linii oznacza kwantyzację (linie poziome) i próbkowanie (linie pionowe). Ponieważ kwantyzacja i próbkowanie - jest wartością dyskretną, powoduje bezpowrotną utratę części informacji i błędy przy próbach przywrócenia sygnału analogowego. Sama definicja sygnału dyskretnego sugeruje, iż sygnał dyskretny nie będzie idealnie oddawać sygnału analogowego. Podczas przetwarzania cyfrowego sygnału można zniwelować błędy kwantyzacji np. stosując dithering. W przypadku ilości bitów przypadającej na próbkę większej niż 16 bitów, błędy są niesłyszalne dla ludzkiego ucha. Dlatego warto stosować maksymalną możliwą wartość kwantyzacji. Dostosowując wyżej wymienione parametry można „oszukać" zmysł słuchu [1, 15]. Czasami zachodzi potrzeba zmiany parametrów dźwięku, które nie są jawnie widoczne w wykresie sygnału np. barwa dźwięku (zbiór częstotliwości). Szybka transformacja Fouriera pozwala wygodnie przekształcać dziedzinę czasu na dziedzinę częstotliwości i odwrotnie [1, 15].

3.2. Problemy związane z DSP

Mimo wzrostu mocy obliczeniowej komputerów i precyzji procesorów, cyfrowe przetwarzanie dźwięku nadal sprawia pewne problemy. Przede wszystkim dokładność odwzorowania jest niewiele poniżej granicy, której przekroczenie pozwoli stwierdzić, iż dźwięku cyfrowego nie da się odróżnić od najwyższej jakości analogowych nagrań. Problemem otwartym pozostaje kwantyzacja na poziomie maksymalnie 64 bitów. Niewiele wskazuje na to, żeby w najbliższym czasie ta sytuacja mogła ulec poprawie [1, 12, 15].

DSP stwarza również problem kosztu obliczeniowego. Obecnie z domowym sprzętem można wykonywać operacje na dźwięku o średniej jakości w czasie rzeczywistym. Stosując np. tylko zmianę barwy dźwięku, można oczekiwać efektu praktycznie natychmiast (tj. czas odpowiedzi jest niezauważalny). Dlatego programowe odtwarzacze muzyki posiadają rozszerzenia umożliwiające stosowanie efektów dźwiękowych. Dźwięk o lepszej jakości wymaga pewnego oczekiwania na zakończenie operacji, bądź też lepszego procesora, aby zakończyć się w rozsądnym czasie. Ponieważ systemy rozpoznawania mowy służą rozpoznaniu konkretnej osoby, poza pierwotnymi zadaniami wymaga się również zastosowanie jako system czasu rzeczywistego [2, 7, 10, 11].

Kolejnym problemem pozostaje jakość sprzętu, który jest używany do przetwarzania sygnału. Najtańszy sprzęt może być mało wydajny i mało precyzyjny. Charakterystyka większości układów elektronicznych powoduje zakłócenia i szumy z powodu jakości użytych materiałów. Niektóre części większych systemów są wyspecjalizowane, np. reduktory szumu białego, czerwonego itp. [2, 7, 10, 11].

Problem jaki można zauważyć najczęściej podczas przetwarzania mowy, jest umiejscowienie źródła dźwięku względem odbiorcy (mikrofonu). Większa odległość mówcy od mikrofonu powoduje, że dźwięk staje się bardziej rozproszony i podatny na zakłócenia. To rozproszenie sprawia, że przejścia między kolejnymi sekcjami dźwięku (np. między głoskami a ciszą) są płynne, więc oddzielenie głosu mówcy od otoczenia jest trudniejsze. Charakterystyka częstotliwościowa takiego dźwięku jest również nieco zbliżona do szumu. Rozwiązaniem na ten problem, jak i na rozpoznanie kierunku dźwięku, jest zastosowanie kilku urządzeń przechwytywania. Mając do dyspozycji dwa mikrofony można stwierdzić, z której strony jest mówca – odpowiednio jeśli dźwięk jest głośniejszy w prawym czy w lewym mikrofonie [22].

4. Systemy rozpoznawania mowy

Celem poniższego rozdziału jest omówienie powszechnie stosowanych systemów rozpoznawania mowy. Z charakterystyki sygnału mowy wynika podobieństwo algorytmów. Parametry opisywanych algorytmów są wynikiem długotrwałych badań na dużej liczbie zebranych wcześniej danych. Gotowe aplikacje działają lepiej jeżeli słowniki mają więcej, z którymi porównywane są nagrane słowa [7, 8, 14].

Między algorytmami automatycznego rozpoznawania mowy istnieją pewnie różnice, według których częściowo da się je sklasyfikować. Istniejące systemy podzielić możemy na takie, których zadaniem jest rozpoznawania mowy ciągłej i takie które mają wykrywać wystąpienia izolowanych słów. Wyróżnić możemy również systemy zależnie od mówcy: dla jednej osoby bądź dla wielu. W pierwszym przypadku algorytmy ukierunkowane są na rozpoznawanie z pewną z góry określoną charakterystyką. W drugim zaś musi zostać spełnione założenie wielu potencjalnych mówców [7, 8, 14].

Kolejnym istotnym podziałem jest klasyfikacja ze względu na wielkość. Są to systemy [5, 14]:

- małe – do około 100 zwrotów-największa skuteczność ze względu na małą ilość zwrotów,
- średnie – co najwyżej 3000 zwrotów-aktualnie są przedmiotem badań i rozwoju,
- duże – 20000 i więcej-cechują się licznymi problemami, szczególnie kiedy mamy zamiar tworzyć uniwersalny system będący w stanie rozpoznawać mowę wielu osób. Trudniejsze, im wydajniej algorytm ma działać. Szczególnie jeśli ma to być system czasu rzeczywistego [3, 10, 14].

4.1. MFCC

Słowo ciężko powtórzyć w taki sam sposób. To jednak nie znaczy, że powtarzane słowa są różne. Nadal noszą one tą samą treść. Dlatego warto się skupić na pewnych cechach wypowiedzi. Oszacowanie podobieństw można uzyskać za pomocą MFCC (ang *Mel Frequency Cepstral Coeficients*) [1, 7, 15, 22].

Dziedzina czasu nie zawsze niesie ze sobą tyle informacji, co dziedzina częstotliwości. To sugeruje przedmiot badań sygnału. W przypadku tego algorytmu, nazwa w pewien sposób odzwierciedla podstawowe działanie i wynik. Jego wynikiem jest wektor cech, na którego podstawie można zaobserwować zmiany sygnału [1, 7, 15, 22].

Dane poddane analizie to pliki wave PCM z typową częstotliwością próbkowania 22050 Hz. Poniżej przedstawiono ogólny schemat postępowania w przypadku MFCC [1, 7, 15, 22]:

- podział dźwięku na nachodzące na siebie ramki o długości około 25 ms,
- obliczenie transformaty Fouriera z oknem Hamminga dla każdej ramki,
- przekształcenie każdego z widm chwilowych do postaci 24 zachodzących na siebie filtrów o kształcie trójkątnym równo rozłożonych na skali mel
- przekształcenie amplitud sygnałów na wyjściach filtrów ze skali liniowej do logarytmicznej
- obliczenie współczynników cepstralnych za pomocą dyskretnej transformaty cosinusowej przeprowadzonej na sygnałach wyjściowych z filtrów, oraz energii sygnału.

Ważne jest rozmieszczenie ramek (nachodzenie) tak, żeby pokrywały istotne fragmenty sygnału. Zarówno długość ramek jak i ich nakładanie można w pewnym zakresie regulować stosownie do pozostałej części aplikacji. Aby uzyskać możliwość dokonywania obserwacji częstotliwości i obliczenia cepstrum, należy najpierw poddać sygnał szybkiej transformacji Fouriera. Specyfika zadania sugeruje użycie okna Hamminga. Wybór tego okna czasowego pozwala na eliminację maksymalnej ilości zniekształceń, co w przypadku innych nie jest możliwe w takim samym stopniu. Dźwięk w domenie częstotliwości poddany jest przekształceniu na skalę mel zgodnie ze wzorem (2). Ze względu na dalsze operacje na dźwięku w skali mel stosuje się bank filtrów. Charakterystyka banku filtrów przypomina okienkowanie trójkątne powtarzane w pewnych przedziałach częstotliwości [1, 22].

W wyniku analizy za pomocą MFCC uzyskuje się wektor około 40 cech w skali son. Wektor ten stanowi charakterystykę dźwięku. Odnajdywanie podobieństw w próbkach dźwięku realizowane jest jako porównanie odległości euklidesowych wektorów cech. Proces ten wyrażony jest wzorem [1, 22]:

$$d(p,q)=d(q,p)=\sqrt{(p_1-q_1)^2+(p_2-q_2)^2+\ldots+(p_n-q_n)^2}=\sqrt{\sum_1^n (q_i-p_i)^2}$$

(3)

gdzie p, q – wektory cech, n – długość wektorów

4.2. Ukryte modele Markowa

W opisie powyższego algorytmu wypada zacząć od definicji łańcuchów Markowa. Niech $Q \neq \emptyset$, będzie skończonym zbiorem (zbiorem stanów). Pewien stan $k_0 \in Q$ jest wyróżniony jako stan początkowy. Łańcuch Markowa zadany jest przez macierz przejść $M=(p_{k,l})k,l \in Q$, która dla $k,l \in Q$ podaje prawdopodobieństwo przejścia ze stanu *k* do stanu l. *M* musi spełniać następujący warunek: dla każdego $k \in Q$ mamy [2]:

$$\sum_{k \in Q} p_{k,l}=1 \qquad (4)$$

Pewien układ, który w każdym momencie może znajdować się w jednym ze stanów $k \in Q$ jest opisywany właśnie łańcuchem Markowa. Układ taki obserwuje się w dyskretnych chwilach czasowych (czyli np. *t* = 0, 1, … n). Przyjmuje się, iż jego początkowy stan to k_0. Jeśli w momencie *t* układ znajduje się w stanie *k*, to w momencie *t* + 1 przechodzi w stan *l* z prawdopodobieństwem *p(k,l)*. Bardzo ważną – wręcz podstawową – cechą łańcuchów Markowa jest fakt, iż obecny stan nie jest zależny od *t* ani historii przejść, tylko od poprzedniego stanu [2].

Ukryte modele Markowa (ang. *Hidden Markov Models*) mają za podstawę definicję łańcuchów Markowa. Niech Σ będzie alfabetem $k \in Q$. Rozważaniu podlegają tutaj łańcuchy Markowa mogące się komunikować z otoczeniem poprzez emitowanie ciągów liter z alfabetu Σ. Mając dany HMM, będący w stanie $k \in Q$, emituje on symbol $x \in \sum$ z prawdopodobieństwem $e_k(x)$ oraz przechodzi do stanu l z prawdopodobieństwem *p(k,l)*. Jeśli w każdym stanie $k_0 \in Q$ ma być emitowany jakiś symbol, to należy przyjąć: $\sum_{x \in \Sigma} e_k(x)=1$

Jeśli dopuszcza się (z pewnym prawdopodobieństwem), że w pewnych stanach nic nie jest emitowane, to przyjmuje się $\sum_{x \in \Sigma} e_k(x) \leqslant 1$. Można przyjąć, że to co da się obserwować, to symbole emitowane przez układ a nie stany wewnętrzne układu (stąd nazwa „ukryte") [2].

Algorytm bazujący na ukrytych modelach Markowa pozwala na uzyskanie podobnych (ewentualnie trochę lepszych) wyników, niż MFCC, jednak nie został on wykorzystany w aplikacji. Na cele demonstracyjne został wybrany prostszy algorytm, a efekty są zadowalające. Dla poprawienia osiągów albo w celu napisania produkcyjnej wersji aplikacji, można posłużyć się również HMM (ang. *Hidden Markov Models*).

5. Aplikacja z wykorzystaniem MFCC

Celem niniejszego rozdziału jest opisanie aplikacji z zaimplementowanym systemem rozpoznawania mowy i omówienie dostępnych narzędzi. System powinien umożliwiać rozpoznanie pojedynczego mówcy na podstawie uprzednio przygotowanych nagrań znajdujących się w słowniku. Aplikacja zawiera praktyczne zastosowanie matematycznego modelu rozpoznawania mowy: rysowanie zgodnie z wydanym poleceniem. Takie zastosowanie systemu automatycznego rozpoznawania mowy pozwala na wykorzystanie szerokiego zakresu poleceń. Ponieważ zbiór poleceń jest stały (tzn. komputer nie ma prawa wykonać nieznanego polecenia), nie ma potrzeby sięgania do innych źródeł po znaczenie słowa, ani próby rozpoznania sensu.

5.1. Problemy i dobór narzędzi

Nie w każdej aplikacji da się zastosować rozpoznawanie mowy. Warto rozważyć system przypominający swoim działaniem konsolę systemową: po wydaniu polecenia i podaniu ewentualnych parametrów, oczekuje się odpowiedzi. Musimy dobrać algorytmy odpowiedzialne za realizację zadań postawionych przed projektem. Należy tu rozważyć przede wszystkim wielkość systemu i wydajność. Za rozmiar informacji w aplikacji rozpoznającej mowę, przyjmujemy rozmiar słownika:

- zajmowana pamięć na dysku,
- ilość mówców do rozpoznania,
- ilość poleceń, których oczekuje aplikacja,
- ilość pozycji w słowniku.

Słownik zajmuje maksymalnie tyle, ile wszystkie jego elementy razem wzięte, ich liczba może być różna. Istnieje rozwiązanie zmniejszenia rozmiaru słownika, wiele słów ma te same rdzenie (podstawa kilku liter) np. liczebniki: „jedenasty" ostatnie 5 liter ma takie same jak „dwunasty". To znaczy, że przyrostek „nasty" można zapisać w słowniku raz, a odpowiednie formatowanie pliku może posłużyć za dopasowanie do reszty. W niniejszej pracy wykorzystano mechanizm serializacji obiektów języka Java. Istotnym czynnikiem wpływającym na rozwój aplikacji jest platforma systemowa i sprzętowa, na której ma działać projekt. Rezygnując z mechanizmów specyficznych dla niektórych systemów, można zyskać uniwersalność. Niniejszy projekt został wykonany w języku Java. Biblioteką odpowiedzialną za graficzny interfejs użytkownika jest Swing. Ten sam kod można skompilować na niemal dowolnym systemie bez uprzedniego przygotowywania projektu. Środowiskiem programistycznym użytym do stworzenia projektu jest Eclipse.

Ważnym elementem projektu są dane wejściowe. Ponieważ rozpoznawanie mowy wiąże się z zapisem dźwięku z mikrofonu, należy zastanowić się jakie parametry powinien mieć ten dźwięk. W ogólnym przypadku przetwarzania sygnału im lepsza jego jakość, tym lepiej. W przypadku rozpoznawania mowy jednak warto obniżyć częstotliwość próbkowania. Pierwszym argumentem przemawiającym za doborem niskiej wartości jest tylko jeden kanał przechwytywania: całość słyszalnego sygnału zmieści się w częstotliwości 22050 Hz. Po drugie: do sygnału mowy nie ma potrzeby używania całego pasma. Można powiedzieć więcej – jest ono zbędne, gdyż nieużywane pasmo może powodować błędy.

Kwantyzacja 8 bitów bez znaku pozwoliła znacząco uprościć schemat przetwarzania. W Javie do przechowywania takiego dźwięku wystarczy tablica typu *byte*. Klasy do strumieniowego odczytu danych z mikrofonu przechwytują „surowe" dane w porcjach po 8 bitów, gdyż to jest najmniejsza możliwa do przechowywanie porcja danych. Z charakterystyki architektury x86 i innych opartych na wymienionej wynika, że w przypadku zmiennej mniejszej, niż 8 bitów miejsce zajmowane przez adres zajmie więcej danych, niż sama zmienna. Kolejnym powodem, dla którego nie może być to mniej, niż 8 bitów, jest szybkość działania: w systemie dwójkowym wyrównanie (adresów) do liczby bitów będącej potęgą dwójki jest najbardziej optymalne. Dlatego decyzja o kwantyzacji pozwala uniknąć nadmiarowych obliczeń podczas nagrywania.

5.2. Schemat i budowa aplikacji

Pisząc aplikację od podstaw, można napotkać kilka problemów:

- jak długo projekt ma być rozwijany – ile wersji ma powstać,
- jakie biblioteki mają być dołączone,
- które części programu mogą/ mają się zmienić.

Java w pewnym sensie „wymusza" pewne usystematyzowanie kodu. Pakiety, specyfikatory dostępu, typy anonimowe i interfejsy pozwoliły na stworzenie modularnej aplikacji. Aby dodać element graficznego interfejsu użytkownika, wystarczy dodać do okna „dziecko", które jest rozszerzeniem dla jednej z podstawowych klas Swing/ AWT. Projekt został podzielony na następujące pakiety, z czego każdy zawiera kilka klas. Poniżej przedstawiono strukturę kodu projektu:

- silentium
  - AudioPreProcessor.java
  - Console.java
  - Dictionary.java
  - DrawCanvas.java
  - FFT.java
  - MFCC.java
  - NewCommandDialog.java
  - Range.java
  - ReducedAudioInputStream.java
  - Silentium.java
  - SoundSystem.java
  - Utils.java
  - Wave.java
  - XMLSerializable.java
- comirva.audio
  - AudoPlayer.java

  - AudioPlaylistPlayer.java
  - XMLSerializable.java
- comirva.audio.util.math
  - CholeskyDecomposition.java
  - EigenvalueDecomposition.java
  - LUDecomposition.java
  - Maths.java
  - Matrix.java
  - NormalizedConvolution.java
  - QRDecomposition.java
  - SingularValueDecomposition.java

Pakiet silentium zawiera główne klasy aplikacji. Pozostałe zawierają klasy biblioteki zwanej CoMIRVA (Collection of Music Information Retrieval and Visualization Applications). Ta biblioteka stanowi zbiór kilku algorytmów pomocnych przy obróbce danych multimedialnych. Zawiera np. algorytm do szybkiej transformacji Fouriera. Niektóre klasy i metody wywoływane przez inne, są niejako pomocnicze. Inne są zależne od systemu: na przykład szybka transformacja Fouriera jest implementowana na różne sposoby, pozwalające na maksymalne wykorzystanie możliwości architektury, więc niektóre metody mogą mieć mniej wspólnego z samym algorytmem, a więcej z danym systemem komputerowym. W tej pracy skupiono się na opisaniu tylko najważniejszych klas i metod. Funkcjonalność taka jak konwersja z typu byte do typu double jest sprawą drugorzędną, nie mającą wpływu na działanie algorytmu (w zasadzie to mają wpływ na szybkość działania, ale bardzo małe ułamki sekund są tu widziane jako zupełny brak różnicy).

Biblioteka Swing domyślnie ma swój specyficzny, niezależny od systemu, wygląd. Kolejnymi zaletami wartymi wspomnienia są:

- duża liczba komponentów,
- dobry MVC (ang. *Model-View-Controller*)
- stabilność, „dojrzałość” projektu (w wersji 1.3.1 Java SE biblioteka została przeniesiona do biblioteki standardowej [21])
- rozszerzalność: budowa bliblioteki pozwala na rozszerzenie niemal dowolnego komponentu (np. JComponent, JButton).

Bibliotece Swing podobnie jak każdej innej, można zarzucić kilka rzeczy, np. dłuższy czas odpowiedzi kontrolek i większe zużycie pamięci RAM niż w przypadku systemowych rozwiązań.

5.3. Realizacja programowa aplikacji

Aplikacja została podzielona na kilka plików. Celem jest opisanie programowej realizacji wymienionych elementów. Pierwszym opisanym elementem jest graficzny interfejs użytkownika i powiązanie go z operacjami wykonywaniami przez aplikację. Kolejnymi wymienianymi elementami będą struktury funkcji odpowiedzialnych za samo wywołanie i operacje. W listingach obejmowano tylko najważniejsze części kodu. Kod niemal całego graficznego interfejsu użytkownika zawarty jest w klasie silentium. W tym pliku znajdują się funkcje realizujące odpowiedź aplikacji na zdarzenia przyciśnięcia przycisku, inicjując tym samym odpowiednie operacje, np. nagrywanie dźwięku. Wykonano je z pomocą obiektów typu ActionListener. W przeciążonej funkcji actionPerformed sprawdzane jest pochodzenie sygnału i wywoływana jest funkcja odpowiednia do przycisku. W poniższych listingach przedstawiono kod realizujący przypisanie zdarzeń do metod i wywoływane funkcje. Kolejne fragmenty kodu przedstawiają operacje wykonywane po naciśnięciu przycisków.

Listing 1. Dowiązanie zdarzeń przyciśnięcia przycisków do funkcji.
Listing 1. *Linking the events of pressing buttons to functions.*

Listing 2. Wczytywanie słownika.
Listing 2. *Loading of the dictionary.*

```
this.buttonsDictionaryListener = new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                if (e.getSource() == _parent.buttonLoadDictionary) {
                        _parent.loadDictionary();
                } else if (e.getSource() == _parent.buttonFindWord) {
                        _parent.recordAndFind();
                } else if (e.getSource() == _parent.buttonAddNewWord) {
                        _parent.addDictionaryPosition();
                } else if (e.getSource() ==        _parent.buttonAddNewWordFromFile) {
                        _parent.addDictionaryPositionFromFile();
                        }
                }
        };
this.buttonsMainListener = new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
                if (e.getSource() == _parent.buttonRecordSound) {
                        _parent.recordCommand();
                } else if (e.getSource() == _parent.buttonClearCanvas) {
                        _parent.canvas.lines.clear();
                }
```

```
public void loadDictionary() {
        int retVal = this.fileChooser.showOpenDialog(this);
        if (retVal == JFileChooser.APPROVE_OPTION) {
                String name = this.fileChooser.getSelectedFile().getAbsolutePath();
                if (this.dictionary == null) {
                        this.dictionary = new silentium.Dictionary(name, 20, 20);
                }
                this.dictionaryPath = name;
...
                this.deserializeDictionary(name);
                for (String current : this.dictionary.wordList) {
                        this.dictionaryPositionsListModel.addElement(current);
                }
                JoptionPane.showMessageDialog(this, "Wczytane pozycje " +
this.dictionary.wordList.size(), "Informacja", JOptionPane.INFORMATION_MESSAGE);
        }
}
```

Listing 3. Użycie serializacji do wczytywania słownika.
Listing 3. *The use of serialization to load the dictionary.*

```
public void deserializeDictionary(String path) {
        try {
                FileInputStream fis = new FileInputStream(path);
                ObjectInputStream ois = new ObjectInputStream(fis);
                this.dictionary = (Dictionary) ois.readObject();
                fis.close();
                ois.close();
        } catch (FileNotFoundException e) {
                e.printStackTrace();
        } catch (IOException e) {
                e.printStackTrace();
        } catch (ClassNotFoundException e) {
                e.printStackTrace();
        }
}
```

Listing 4. Użycie mechanizmu serializacji do zapisu słownika.
Listing 4. The use of serialization mechanism to save the dictionary.

```
public void serializeDictionary() {
        try {
                if (this.dictionary == null) {
                        JOptionPane.showMessageDialog(this, "Nie można zapisać słownika",
                                        "Informacja", JOptionPane.ERROR_MESSAGE);
                        return;
                }
                FileOutputStream fos = new FileOutputStream(System.getProperty("user.home")
                                + System.getProperty("file.separator") +
"SilentiumDictionaryDump.bin");
                ObjectOutputStream oos = new ObjectOutputStream(fos);
                oos.writeObject(this.dictionary);
                oos.flush();
                fos.close();
                oos.close();
        } catch (FileNotFoundException e) {
                e.printStackTrace();
        } catch (IOException e) {
                e.printStackTrace();
        }
}
```

Listing 5. Uzyskanie wektora cech ze strumienia dźwięku.
Listing 5. Obtaining of the feature vector from the stream of sound.

```
public double[] processWindow(double[] window, int start) throws IllegalArgumentException {
        int fftSize = (windowSize / 2) + 1;
        if (start < 0)
                throw new IllegalArgumentException("start must be a positve value");
        if (window == null || window.length - start < windowSize)
                throw new IllegalArgumentException("the given data array must not be a null value
and must contain data for one window");
        for (int j = 0; j < windowSize; j++)
                buffer[j] = window[j + start];
        normalizedPowerFFT.transform(buffer, null);
        Matrix x = new Matrix(buffer, windowSize);
        x = x.getMatrix(0, fftSize - 1, 0, 0);
        x = melFilterBanks.times(x);
        double log10 = 10 * (1 / Math.log(10));
        x.thrunkAtLowerBoundary(1);
        x.logEquals();
        x.timesEquals(log10);
        x = dctMatrix.times(x);
        return x.getColumnPackedCopy();
}
```

5.4. Wyniki działania aplikacji

Aby aplikacja spełniała stawiane przed nią zadania, należy sprawdzać zgodność jej działania z oczekiwaniami. Musi to się odbywać niemal na każdym etapie pisania. Testowanie aplikacji podczas pisania dotyczyło głównie na badaniu poprawności reakcji programu na polecenia użytkownika. Test działania najważniejszego elementu, czyli rozpoznawania mowy, odbywał się pod niemal na końcu projektu, ponieważ wymagało to przygotowania paru innych elementów np. zbieranie próbek głosu z mikrofonu.
W procedurze testowej stworzono słownik różnymi nagraniami – dostępnymi w aplikacji poleceniami. Następnie sprawdzona została funkcja wyszukiwania słowa, które powinno istnieć w słowniku. Wyszukiwanie polegało na wyznaczeniu odległości euklidesowych między wektorami MFCC dla każdego słowa i wyborze najmniejszej odległości.

Tab. 1. Skuteczność działania mechanizmu rozpoznawania.
Tab. 1. The effectiveness of the recognition mechanism.

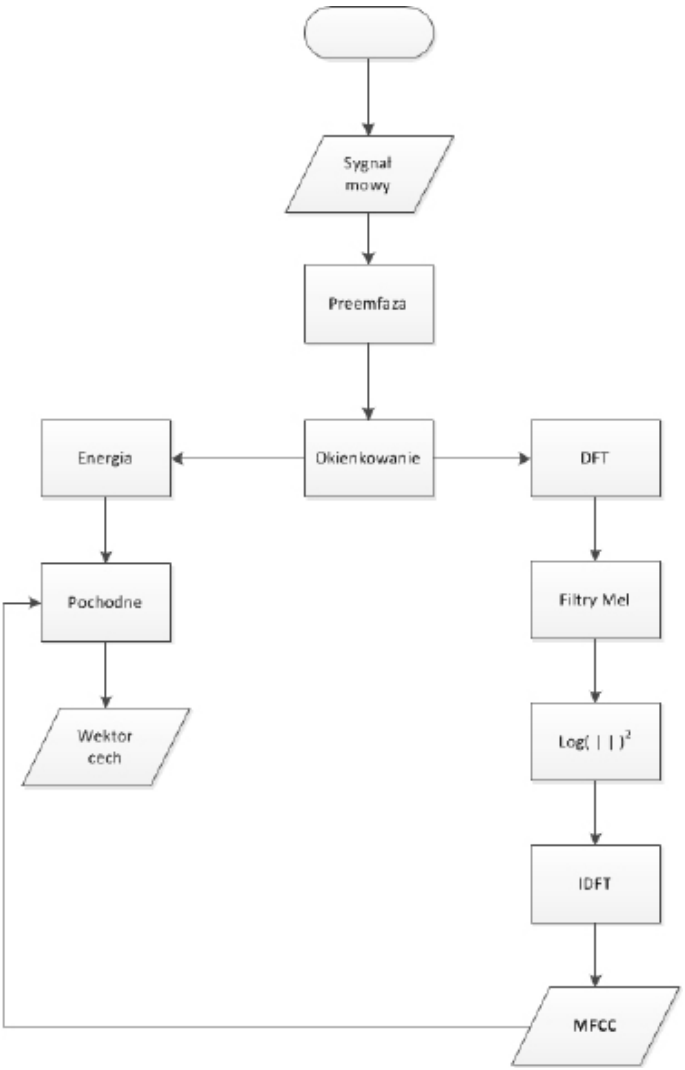
Słowo	Próby	Rozpoznania prawidłowe
prawa	25	20 (80%)
lewa	40	31 (~77%)
cofnij	35	22 (~63%)
góra	20	13 (65%)
wyczyść	36	23 (~63%)
zamknij	45	40 (~88%)

5.5 Zestawienie wydajności MFCC - Działanie programu

Do przeprowadzonych testów został stworzony słownik w kilku rozmiarach. Pomiary zostały dokonane dla operacji wyszukiwania w słowniku słowa nagranego mikrofonem. Na wyszukiwanie składają się następujące operacje:

- skalowanie dźwięku
- wyliczanie MFCC dla nagranego słowa
- porównanie z pozycjami słownika: odległości euklidesowe

Schemat działania programu opartego o MFCC przedstawiono na Ryc. 4.



Ryc. 4. Schemat działania algorytmu wyznaczania MFCC.
Fig. 4. *The diagram of the MFCC determining algorithm.*

Porównanie według ilości wątków

W poniższej tabeli przedstawiono czas wykonania algorytmu w zależności od zasobów użytych na komputerze Sun Constellation System Halo2. Dla przykładu operację porównania powtórzono kilka razy.

Tab. 2. Czas realizacji zrównoleglonego algorytmu MFCC na komputerze Sun Constellation System Halo2 (czasy podane w mikrosekundach).
Tab. 2. Duration time of the parallelized MFCC algorithm on a Sun Constellation System Halo2 computer (times in microseconds).

Próba	1 wątek	2 wątki	4 wątki	16 wątków	32 wątki
1	45913039	22899451	9981095	2849774	1481066
2	48639417	23181627	10573786	3648998	1569013
3	51054355	24811597	11298702	3668891	1646915
4	52799836	25542779	11478225	3277231	1703221
5	54441505	25924526	11865109	3471128	1756178
6	101170839	48676590	21993660	6279569	3263575
7	116008978	55242370	28219343	7200957	3742225
8	120149721	57214152	26119504	7457569	3875797
9	124332854	59206120	27028881	7717212	4019737

Porównanie według maszyn

Zestawienie zawiera średni czas działania programu w zależności od rozmiaru słownika i użytego systemu.

Tab. 3. Porównanie wydajności trzech różnych komputerów: IBM Blue Gene/Q, Sun Constellation System Halo2 oraz IBM Power 775 podczas realizacji zrównoleglonego algorytmu MFCC. Wykorzystano dwa wątki (czasy podane w mikrosekundach).
Tab. 3. *Compare of the performance of three different computers: IBM Blue Gene/Q, Sun Constellation System Halo2 and IBM Power 775 during the realisation of the MFCC parallelized algorithm. Uses two threads* (*times in microseconds*)

Rozmiar słownika	IBM Blue Gene/Q	Sun Constellation System Halo2	IBM Power 775
256	55968900	34810729	45638649
512	96382944	59793289	84824320
1024	147881027	104780294	132123888
2048	313507777	201178164	265569014
4096	732040659	362120695	589563211

6. Wnioski

W niniejszej pracy zaprezentowano działanie tylko jednego spośród algorytmów rozpoznawania mowy. Wybór odpowiedniego algorytmu nie zawsze jest prosty. Mimo znacznych postępów idealny algorytm nie istnieje. Z tego względu warto rozważać wybór pod konkretny projekt i konkretne zastosowania. Jak z każdym większym projektem, należy przed rozpoczęciem działań zastanowić się nad próbami ich rozwiązania. Może się okazać, że nie wszystkie problemy wystąpią naraz i nie każdy będzie tak dotkliwy, żeby znacząco utrudnić tworzenie bądź też używanie aplikacji. Jeśli system ma rozpoznawać konkretne słowa bądź konkretnego mówcę, to nie trzeba martwić się rozróżnianiem mówców [1, 15, 22].
Nowe technologie mogą budzić zarówno podziw i fascynację, jak i obawy. Komputery i telefony są najczęstszymi środkiem komunikacji. Ludzie starsi, chorzy czy w jakikolwiek inny sposób pozbawieni możliwości korzystania z komputerów nie mogąc komunikować się z innymi za ich pomocą, tacy ludzie tracą kontakt ze znajomymi i rodziną. Projektując nowe algorytmy należy brać pod uwagę takie problemy i starać się dostosować aplikację do jak największej liczby korzystających [13, 25, 26].
Aplikacja spełniła swoje założenia. Graficzny interfejs użytkownika pozwolił na wygodne ukazanie działania takiej aplikacji. Wykorzystany język programowania pozwolił na możliwe szerokie udostępnienie aplikacji. Modularna budowa projektu stanowi tutaj dobrą podstawę do dalszego rozwoju.

Podziękowania

Obliczenia wykonano w Interdyscyplinarnym Centrum Modelowania Matematycznego i Komputerowego (ICM) Uniwersytetu Warszawskiego w ramach grantu obliczeniowego nr G55-11.

Literatura (References)

[1] X. Huang, Spoken Language Processing - A Guide to Theory, Algorithm, and System Development, Prentice Hall PTR 2001.
[2] A. M. Wiśniewski, Automatyczne rozpoznawanie mowy bazujące na ukrytych modelach Markowa – problemy i metody, Biuletyn Instytutu Automatyki i Robotyki WAT nr 12, 2001.
[3] A. Viterbi, Eror bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Information Theory, vol IT-13, pp 260-269, 1967
[4] J. Schalkwyk, P. Hoson, E. Kasier, K. Shobaki, CSLU -HMM: The CSLU Hidden Markov Modelling Environment, Center of Spoken Language Understanding, Oregon Graduate Institute of Science & Technology
[5] F. Fissore, E. Giachin, P. Laface, P. Massafra, Using Grammars in forward and backward search, Proceedings Eurospeech 9, Berlin 1993.
[6] M. Schedl, http://www.cp.jku.at/people/schedl/Research/Development/CoMIRVA/webpage/CoMIRVA.html, stan z dnia 25.01.2013.
[7] Carnegie Mellon University, http://cmusphinx.sourceforge.net/, stan z dnia 25.01.2013.
[8] Carnegie Mellon University, http://www.speech.cs.cmu.edu/, stan z dnia 25.01.2013.
[9] NVIDIA Corporation, http://www.nvidia.pl/object/cuda_home_new_pl.html, stan z dnia 25.01.2013
[10] Massachusetts Institute of Technology, http://mit.edu, stan z dnia 25.01.2013.
[11] Alcatel- Lucent (Bell Labs) http://www.alcatel-lucent.com, stan z dnia 25.01.2013.
[12] G. Moore, Cramming more components onto integrated circuits, Electronics, Volume 38, Number 8, April 19, 1965.
[13] On Board PR Ecco Network, http://www.onboard.pl/data/file/pdf/raport__swiadomosc_polakow_w_rzeczywistosci_cyfrowej.pdf, stan z dnia 25.01.2013
[14] K. R. Farrell, R. J. Mammone, K. T. Assaleh, Speaker Recognition Using Neural Networks and Conventional Classifiers, IEEE TRANSACTIONS ON SPEECH AND AUDIO PROCESSING, VOL. 2, NO. 1, PART 11, JANUARY 1994.
[15] T. P. Zieliński, Cyfrowe Przetwarzanie Sygnałów – Od Teorii do Zastosowań, Wydawnictwa Komunikacji i Łączności, Wydanie 1, Warszawa 2005
[16] Rada Języka Polskiego, http://www.rjp.pan.pl, stan z dnia 25.01.2013.
[17] Eclipse Fundation, http://www.eclipse.org, stan z dnia 04.02.2013.
[18] Oracle Corporation, http://www.oracle.com, stan z dnia 04.02.2013.
[19] R.J. Marks II, Introduction to Shannon Sampling and Interpolation Theory, Spinger-Verlag, New York 1991.
[20] W. Chen, https://blogs.oracle.com/Swing/entry/awt_swt_swing_java_gui3, stan z dnia 05.02.2013.
[21] Oracle Corporation, Swing 1.3 features http://docs.oracle.com/javase/1.3/docs/relnotes/features.html, stan z dnia 05.02.2013.
[22] M. L. Seltzer, Microphone Array Processing for Robust Speech Recognition, Carnegie Mellon University, Pittsburgh 2003.
[23] Microsoft, http://msdn.microsoft.com/pl-pl/ms348103.aspx, stan z dnia 07.02.2013.
[24] Microsoft, MSDN http://msdn.microsoft.com/en-us/vstudio/hh341490.aspx, stan z dnia 07.02.2013.
[25] Social Press, http://socialpress.pl/2013/02/sotrender-podsumowuje-2012-rok-na-polskim-facebooku-popularnosc-marek-rosnie-w-bardzo-szybkim-tempie/#, stan z dnia 08.02.2013.
[26] Gazeta Wyborcza, http://wyborcza.biz/biznes/1,101562,6289081,Nokia_bierze_Skype_a_na_poklad.html, stan z dnia 08.02.2013.