ISSN 0239–8044

1
1996

Techniki Komputerowe

BIULETYN INFORMACYJNY

INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA 1996

Techniki Komputerowe

BIULETYN INFORMACYJNY

Rok XXXI, Nr 1, 1996

INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA 1996

Wydaje:

INSTYTUT MASZYN MATEMATYCZNYCH
UL. KRZYWICKIEGO 34
02-798 WARSZAWA
TEL. 621.84.41, TLX 81.78.80, FAX 629.92.70
E-MAIL imasmat@atos.warman.com.pl

Wydanie publikacji dofinansowane
przez Komitet Badań Naukowych

Druk: Zakład Poligrafii Ośrodka Przetwarzania Informacji, al. Niepodległości 186

TECHNIKI KOMPUTEROWE

Rok XXXI Nr 1 1996

Spis treści

ANDRZEJ ABRAMOWICZ
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Dedukcja w systemach informacyjnych zawierających niepełną informację
On deduction in Nondeterministic Knowledge Representation Systems

Streszczenie

W artykule przedstawiono aparat dedukcyjny służący do wnioskowania w systemach informacyjnych zawierających niepełną - w pewnym sensie - informację. Zastosowane konstrukcje nawiązują do logiki klasycznej, jednak zaproponowany język zawiera operatory teoriomnogościowe, co umożliwia bezpośredni dostęp do podstawowych elementów informacji. Składnia i semantyka języka, a także metody wnioskowania, są konsekwencją rozważań dotyczących efektywnego wykorzystania pewnej klasy systemów eksperckich. W Dodatku przedstawiono ważniejsze pojęcia dotyczące systemów informacyjnych oraz podstawowe własności tych systemów.

Abstract

This paper describes the deduction mechanism used in reasoning based on incomplete information. The applied language has the ability to manipulate and access the basic components of information. The language and its reasoning methodology have been derived from an efficient use of a class of expert systems. The Appendix contains the glossary of the pertinent terms and properties of the information systems.

1. Charakterystyka problemu

W wielu dziedzinach zastosowań zasadniczą rolę odgrywa umiejętność posługiwania się nieprecyzyjną informacją. Jako przykłady można podać systemy eksperckie, systemy wspomagające stawianie diagnoz w medycynie, analizę procesów przemysłowych, czy bazy danych w kryminalistyce zawierające - z natury rzeczy - nieprecyzyjną informację. Umiejętność posługiwania się takim rodzajem informacji sprowadza się w istocie do problemu właściwego opisu informacji oraz zastosowania odpowiednich metod wnioskowania.

Obszarem naszych zainteresowań będzie szczególny rodzaj nieprecyzyjnej informacji, a mianowicie informacja niepełna. Załóżmy, że obserwujemy pewne zjawisko, np. proces technologiczny, na tyle skomplikowany, że nie można go opisać

równaniami. W takim przypadku pozostaje jedynie obserwacja działania systemu poprzez rejestrowanie następujących faktów: jeśli wartości $x_1,...,x_n$ parametrów wejściowych $a_1,...,a_n$ spełniają warunki $P_1,...,P_n$, to odpowiednie wartości $y_1,...,y_m$ parametrów wyjściowych $b_1,...,b_m$ spełniają odpowiednio warunki $Q_1,...,Q_m$. Można więc każdorazowo wskazać zbiory potencjalnych wartości parametrów wejściowych i odpowiadające im zbiory wartości parametrów wyjściowych, nie można natomiast ustalić dokładnych wartości parametrów. Sytuacja taka może wynikać np. z błędu pomiaru. Przykładów jest wiele: od badania kolejnych cykli procesu wielkopiecowego w hucie, po rejestrowanie odpowiedzi organizmu na podawanie leków.

Zjawiska takie można opisać wykorzystując pojęcie Systemu Reprezentacji Wiedzy (SRW). W „klasycznym" Systemie Reprezentacji Wiedzy (systemie informacyjnym), własności obiektów są wyrażone poprzez atrybuty (np. KOLOR) i wartości atrybutów (np. ZIELONY). W systemie zawierającym niepełną - w powyższym sensie - informację mamy do czynienia z wartościami atrybutów będącymi zbiorami (np. wartością atrybutu KOLOR jest zbiór {BIAŁY, NIEBIESKI, CZERWONY}). Niepełność informacji polega więc na tym, że nie można podać konkretnej wartości atrybutu, można powiedzieć jedynie tyle, że należy ona do pewnego ustalonego zbioru. W dalszym ciągu będziemy zakładać, że SRW zawiera właśnie ten typ informacji, zwany informacją niedeterministyczną lub przybliżoną. Podamy teraz formalną definicję SRW. SRW zawierający niedeterministyczną informację jest czwórką

$$\boldsymbol{S} = (U, AT, (VAL_a)_{a \in AT}, f),$$

gdzie U, AT i VAL_a, dla każdego $a \in AT$, są odpowiednio niepustym zbiorem obiektów, atrybutów i wartości atrybutów,

$$f: U \times AT \to 2^{VAL},$$

gdzie $VAL = \bigcup_{a \in AT} VAL_a$ jest funkcją całkowitą taką, że $f(x,a) \subset VAL_a$ dla każdego $x \in U$, $a \in AT$. Dla ustalonego a, zbiór $f(x,a)$ nazywamy uogólnioną wartością atrybutu a.

Dla każdego $x \in U$ definiujemy funkcję

$$f_x: AT \to 2^{VAL},$$

którą nazywamy (niedeterministyczną) informacją o obiekcie x, taką, że $f_x(a) = V$ wtedy i tylko wtedy, gdy $f(x,a) = V$, dla $V \subset VAL_a$.

U i AT są zbiorami skończonymi, $AT = C \cup D$, gdzie C, D są niepustymi, rozłącznymi zbiorami atrybutów - odpowiednio - warunków i decyzji.

Niech A będzie podzbiorem AT. Definiujemy relację binarną $inf(A)$ (inkluzji informacyjnej ze względu na zbiór A) w sposób następujący:

$(x,y) \in inf(A)$ wtedy i tylko wtedy, gdy $f(x,a) \subset f(y,a)$ dla każdego $a \in A$

oraz relację binarną $sim(A)$ (podobieństwa informacyjnego ze względu na zbiór A):

$(x,y) \in sim(A)$ wtedy i tylko wtedy, gdy $f(x,a) \cap f(y,a) \neq \emptyset$ dla każdego $a \in A$.

W przypadku opisanych powyżej zjawisk, atrybuty systemu $\boldsymbol{S}$ są odpowiednikami parametrów, a każda informacja f_x w $\boldsymbol{S}$ jest interpretowana w następujący sposób: jeśli wartości atrybutów warunków są odpowiednio elementami zbiorów $f_x(a)$ dla każdego $a \in C$, to wartości atrybutów decyzji są odpowiednio elementami zbiorów $f_x(b)$ dla każdego $b \in D$. Oczywiście każdy obiekt ze zbioru U może być traktowany jako

odnotowany fakt (doświadczenie). Łatwo zauważyć, że w S powinien być spełniony następujący warunek:

jeśli $(x,y) \in sim(C)$ to $(x,y) \in sim(D)$ dla wszystkich $x, y \in U$.

Warunek ten wynika wprost z implikacyjnego charakteru informacji f_x w S i wyraża (rozumianą intuicyjnie) niesprzeczność systemu.

Mając dany System Reprezentacji Wiedzy utworzony na podstawie doświadczeń, można sformułować następujące oczywiste pytania, będące istotą tzw. problemu namiarowania:

- czy jeśli wartości parametrów wejściowych spełniają warunki $P_1,...,P_n$, to należy oczekiwać, że odpowiednie wartości parametrów wyjściowych będą spełniały warunki $Q_1,...,Q_m$;
- jakie warunki będą spełniały wartości parametrów wyjściowych, jeśli parametry wejściowe spełniają $P_1,...,P_n$;
- jakie warunki muszą spełniać parametry wejściowe, aby wartości parametrów wyjściowych spełniały $Q_1,...,Q_m$.

Dla ustalonego systemu $S = (U, AT, (VAL_a)_{a \in AT}, f)$, definiujemy indeksowaną rodzinę $(G_a)_{a \in AT}$ podzbiorów zbioru VAL, taką, że $G_a \subset VAL_a$ dla każdego $a \in AT$, zwaną namiarem. Teraz możemy sformułować pierwsze z powyższych pytań jako warunek:

istnieje $x \in U$ takie, że $G_a \subset f(x,a)$ dla każdego $a \in C$ i $f(x,b) \subset G_b$ dla każdego $b \in D$.

Zauważmy jednak, że nawet jeśli powyższy warunek nie jest spełniony, to odpowiedź na pytanie nie musi być jednoznacznie negatywna. Załóżmy bowiem, że spełniony jest słabszy warunek:

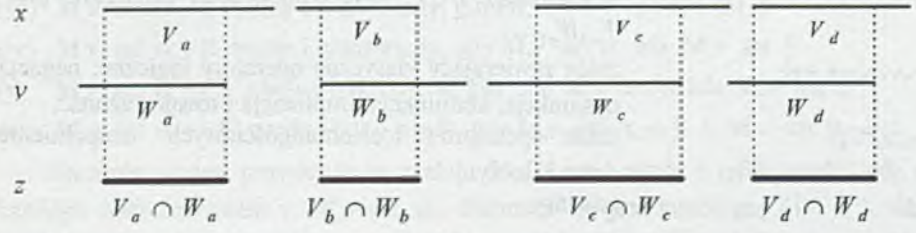
istnieje $x \in U$ takie, że $f(x,a) \cap G_a \neq \emptyset$ dla każdego $a \in AT$.

Oznacza to, że jest możliwe, że jeśli wartości atrybutów warunku są elementami G_a, dla $a \in C$, to wartości atrybutów decyzji są elementami G_b, $b \in D$.

Łatwo zauważyć, że system S nie zawiera explicite pełnej informacji uzyskanej na podstawie przeprowadzonych i zarejestrowanych w S doświadczeń. Weźmy pod uwagę następującą informację:

$$f_z(a) = f_x(a) \cap f_y(a) \text{ dla każdego } a \in AT.$$

$f_z(a)$ można traktować jako informację o (hipotetycznym) obiekcie z. f_z może być zatem traktowana jako wniosek wyprowadzony (w sposób nieformalny) z przesłanek f_x, f_y. Możliwość przeprowadzenia tego typu wnioskowania jest konsekwencją implikacyjnej interpretacji informacji w S. Powyższe rozumowanie ilustruje rysunek, na którym a, b są atrybutami warunków, c i d są atrybutami decyzji.



Jeśli C i D są zbiorami jednoelementowymi, intuicyjnie poprawny jest również inny wniosek: $f_z(a) = f_x(a) \cup f_y(a)$ dla każdego $a \in AT$.

Rozważania na poziomie modelu prowadzą więc do wniosku, że aby odpowiedzieć poprawnie na sformułowane powyżej pytania należy wziąć pod uwagę nie tylko informacje zawarte explicite w bazie wiedzy, ale i pewne wnioski, które można z nich wyprowadzić. Wynika stąd potrzeba sformułowania aparatu dedukcyjnego, który pozwoli na logicznie poprawne wnioskowanie z faktów zarejestrowanych w bazie wiedzy na podstawie wcześniej przeprowadzonych doświadczeń. Jest to tym bardziej istotne, że trudno wyobrazić sobie bazę wiedzy zawierającą wszystkie możliwe fakty nawet w przypadku, gdy dziedziny atrybutów są zbiorami skończonymi. Badania tego typu mają zasadnicze znaczenie dla rozwoju systemów eksperckich wykorzystywanych do wspomagania podejmowania decyzji w wielu dziedzinach - od techniki po medycynę. Naszym celem jest więc zaprojektowanie języka opisu rozważanych zjawisk o odpowiedniej składni i o semantyce, w której model związany jest w naturalny sposób z Systemem Reprezentacji Wiedzy zawierającym niedeterministyczną informację, a także sformułowanie stosownych reguł wnioskowania. Do badania systemów informacyjnych zawierających niedeterministyczną informację używano m. innymi aparatu logik modalnych. W naszym przypadku sytuacja jest jednak inna. W „klasycznym" systemie informacja jest po prostu funkcją (wierszem tabelki), zaś atrybutów nie dzieli się na podzbiory zawierające atrybuty warunków i decyzji. W tej sytuacji łatwo można wprowadzić do języka operatory modalne (konieczność i możliwość), podczas gdy w modelu definiowane są odpowiadające tym operatorom relacje inkluzji i podobieństwa informacyjnego jako relacje dostępności. Takie rozwiązanie nie może być zaadaptowane w przypadku, gdy zbiór atrybutów jest podzielony, na informację narzucona jest pewna interpretacja (informacja ma charakter implikacyjny), a dodatkowo istnieje konieczność wprowadzenia do języka operatorów teoriomnogościowych.

2. Składnia języka

Formuły języka definiujemy używając symboli z następujących, niepustych, parami rozłącznych i co najwyżej przeliczalnych zbiorów:

T - zbiór typów reprezentujących atrybuty,

$VAR = \bigcup_{a \in AT} VAR_a$ - zbiór zmiennych, gdzie VAR_a jest zbiorem zmiennych typu a. Dla każdego $a \in T$, zmienne typu a będą oznaczane przez $x_a, y_a, \ldots$,

$CPVAL = \bigcup_{a \in AT} CPVAL_a$ - zbiór stałych reprezentujących uogólnione wartości atrybutów, gdzie $CPVAL_a$ jest zbiorem stałych typu a. Dla każdego $a \in T$, stałe typu a będą oznaczane przez $V_a, W_a, \ldots$,

$\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$ - zbiór zawierający klasyczne operatory logiczne: negacja, dysjunkcja, koniunkcja, implikacja i równoważność,

$\{-, \cup, \cap\}$ - zbiór operatorów teoriomnogościowych: uzupełnienie, suma i iloczyn,

$\{(,)\}$ - nawiasy.

Zdefiniujemy teraz zbiór (*EPVAL*) wyrażeń interpretowanych jako uogólnione wartości atrybutów. $EPVAL = \bigcup_{a \in AT} EPVAL_a$, gdzie $EPVAL_a$ jest najmniejszym zbiorem wyrażeń typu a spełniającym następujące warunki (dla uproszczenia uogólnione wartości atrybutów będą oznaczane w ten sam sposób jak odpowiednie stałe):

$CPVAL_a \subset EPVAL_a$;

jeśli $V_a \in EPVAL_a$ to $-V_a \in EPVAL_a$;

jeśli $V_a, W_a \in EPVAL_a$ to $V_a \cap W_a, V_a \cup W_a \in EPVAL_a$.

Zbiór formuł *FOR* jest najmniejszym zbiorem takim, że

$(x_a, V_a) \in FOR$ dla każdego $a \in T$, $x_a \in VAR_a$ i $V_a \in EPVAL_a$;

jeśli $\alpha, \beta \in FOR$, to $\neg\alpha, \alpha \vee \beta, \alpha \wedge \beta, \alpha \rightarrow \beta, \alpha \leftrightarrow \beta \in FOR$.

Formula atomowa postaci (x_a, V_a) wyraża fakt, że wartość atrybutu a jest elementem uogólnionej wartości a.

3. Semantyka

Jako model przyjmujemy system

$$\mathbf{M} = (AT, (VAL_a)_{a \in AT}, m),$$

gdzie AT jest niepustym zbiorem atrybutów, VAL_a jest niepustym zbiorem wartości atrybutu a, m jest funkcją przyporządkowującą znaczenie typom i wyrażeniom $m: T \cup EPVAL \rightarrow AT \cup 2^{VAL}$, gdzie $VAL = \bigcup_{a \in AT} VAL_a$, spełniającą następujące warunki:

(i) $m(t) \in AT$ dla $t \in T$.

(ii) $m(V_a) \subset VAL_{a'}$ dla $V_a \in CPVAL_a$, gdzie $a' = m(a)$

(iii) $m(-V_a) = VAL_{a'} - m(V_a)$

(iv) $m(V_a \cap W_a) = m(V_a) \cap m(W_a)$

(v) $m(V_a \cup W_a) = m(V_a) \cup m(W_a)$.

Wartościowanie definiujemy jako funkcję

$$v: VAR \rightarrow VAL$$

taką, że $v(x_a) \in VAL_{a'}$, gdzie $a' = m(a)$.

Dla danego modelu **M** i wartościowania v mówimy, że formula α jest spełniona przez v w **M** (**M**,v sat α), jeśli spełnione są następujące warunki:

(i) **M**,v sat (x_a, V_a) wtedy i tylko wtedy, gdy $v(x_a) \in m(V_a)$

(ii) **M**,v sat $\neg\alpha$ wtedy i tylko wtedy, gdy non **M**,v sat α

(iii) **M**,v sat $\alpha \wedge \beta$ wtedy i tylko wtedy, gdy **M**,v sat α i **M**,v sat β

(iv) **M**,v sat $\alpha \vee \beta$ wtedy i tylko wtedy, gdy **M**,v sat α lub **M**,v sat β

(v) **M**,v sat $\alpha \rightarrow \beta$ wtedy i tylko wtedy, gdy non **M**,v sat α lub **M**,v sat β

(vi) **M**,v sat $\alpha \leftrightarrow \beta$ wtedy i tylko wtedy, gdy **M**,v sat $\alpha \rightarrow \beta$ i **M**,v sat $\beta \rightarrow \alpha$

Formula α jest prawdziwa w modelu **M** ($\models_{\mathbf{M}} \alpha$) wtedy i tylko wtedy, gdy dla każdego wartościowania v, **M**,v sat α. Formuła α jest tautologią ($\models \alpha$) wtedy i tylko wtedy, gdy jest prawdziwa w każdym modelu. Zbiór formuł Φ jest prawdziwy

w modelu **M**, jeśli każda formuła $\alpha \in \Phi$ jest prawdziwa w **M**. Zbiór Φ jest spełnialny wtedy i tylko wtedy, gdy istnieje model **M** i wartościowanie v takie, że $\mathbf{M},v$ *sat* α dla każdej formuły $\alpha \in \Phi$. Formuła α jest semantyczną konsekwencją zbioru Φ ($\Phi \models \alpha$) wtedy i tylko wtedy, gdy dla każdego modelu **M** formuła α jest prawdziwa w **M**, jeśli tylko Φ jest prawdziwy w **M**.

Łatwo można udowodnić następujące fakty:

Lemat 3.1

(a) $\mathbf{M},v$ *sat* $(x_a,-V_a)$ *wtedy i tylko wtedy, gdy* $\mathbf{M},v$ *sat* $\neg(x_a,V_a)$

(b) $\mathbf{M},v$ *sat* $(x_a,V_a \cap W_a)$ *wtedy i tylko wtedy, gdy* $\mathbf{M},v$ *sat* $(x_a,V_a) \wedge (x_a,W_a)$

(c) $\mathbf{M},v$ *sat* $(x_a,V_a \cup W_a)$ *wtedy i tylko wtedy, gdy* $\mathbf{M},v$ *sat* $(x_a,V_a) \vee (x_a,W_a)$

(d) $\models_{\omega} ((x_a,V_a) \rightarrow (x_a,W_a))$ *wtedy i tylko wtedy, gdy* $m(V_a) \subset m(W_a)$

Strukturę dedukcyjną języka określimy w zwykły sposób, specyfikując aksjomaty i reguły wnioskowania oraz definiując operację konsekwencji syntaktycznej. Aksjomatyzacja nawiązuje w naturalny sposób do logiki klasycznej. Aksjomaty (A1)-(A4) są aksjomatami logiki klasycznej, pozostałe charakteryzują formuły zawierające operatory teoriomnogościowe w związku z 3.1 (a), (b), (c):

(A1) $\alpha \rightarrow (\beta \rightarrow \alpha)$

(A2) $(\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma))$

(A3) $\alpha \rightarrow (\neg\alpha \rightarrow \beta)$

(A4) $(\neg\alpha \rightarrow \alpha) \rightarrow \alpha$

(A5) $(x_a,-V_a) \leftrightarrow \neg(x_a,V_a)$

(A6) $(x_a,V_a \cap W_a) \leftrightarrow ((x_a,V_a) \wedge (x_a,W_a))$

(A7) $(x_a,V_a \cup W_a) \leftrightarrow ((x_a,V_a) \vee (x_a,W_a))$

Operatory zdaniowe $\vee, \wedge, \leftrightarrow$ są definiowane w zwykły sposób poprzez implikację i negację.

Regułami wnioskowania są:

$$\frac{\alpha,\ \alpha \rightarrow \beta}{\beta}\ \text{(Modus Ponens)} \qquad \frac{\alpha(x_a)}{\alpha(y_a)}$$

Oczywiście aksjomaty są tautologiami i reguły wnioskowania prowadzą od tautologii do tautologii.

Wnioskowanie formuły α ze zbioru formuł Φ definiujemy jako skończony ciąg formul, z których każda jest aksjomatem lub formułą ze zbioru Φ, lub też jest formułą otrzymaną z wcześniej wygenerowanych formuł przy zastosowaniu reguły wnioskowania, natomiast α jest ostatnią formułą w ciągu. Powiemy, że formuła α jest wnioskowalna z Φ ($\Phi \vdash \alpha$) jeśli istnieje wnioskowanie α z Φ. Formuła α jest twierdzeniem ($\vdash \alpha$), jeśli jest wnioskowalna jedynie z aksjomatów. Zbiór formul Φ jest niesprzeczny, jeśli formuła postaci $\alpha \wedge \neg\alpha$ nie jest wnioskowalna z Φ. Zbiór formuł jest sprzeczny jeśli nie jest niesprzeczny.

Dowód twierdzenia o pełności jest wariantem dowodu twierdzenia o pełności w logice klasycznej. W dalszych rozważaniach użyteczny będzie łatwy do udowodnienia lemat:

Lemat 3.2

(a) *Jeśli* $\Phi\vdash \alpha\to\beta$ i $\Phi\vdash \gamma\to\delta$, *to* $\Phi\vdash (\alpha\wedge\gamma)\to(\beta\wedge\delta)$

(b) *Jeśli* $\Phi\vdash \alpha\to\beta$ i $\Phi\vdash \alpha\to\gamma$, *to* $\Phi\vdash \alpha\to(\beta\wedge\gamma)$

(c) *Jeśli* $\Phi\vdash \alpha\to\beta$ i $\Phi\vdash \beta\to\gamma$, *to* $\Phi\vdash \alpha\to\gamma$

Mając dany model $\mathbf{M}=(AT,(VAL_a)_{a\in AT},m)$ możemy teraz zdefiniować zbiór wyrażeń, które nazwiemy **M**-formułami, otrzymanych z formuł naszego języka przez przyporządkowanie wartości stałym, określone przez odwzorowanie *m*. Pojęcie wnioskowalności można w naturalny sposób rozszerzyć na te formuły. Powiemy, że **M**-formuła jest wnioskowalna ze zbioru Φ **M**-formuł jeśli można ją otrzymać z **M**-formuł mających postać aksjomatów lub elementów zbioru Φ poprzez sukcesywne stosowanie regul wnioskowania. Będziemy używali tej samej notacji co powyżej, tak więc napiszemy $\Phi\vdash \alpha$ jeśli **M**-formuła α jest wnioskowalna ze zbioru Φ **M**-formul.

Przykład

Niech

$$S=(\{x,y\},\{a,b,c,d\},(VAL_t)_{t\in AT},f)$$

będzie SRW takim, że $f(x,t)=V_t$, $f(y,t)=W_t$ dla $t\in AT$, $C=\{a,b\}$, $D=\{c,d\}$ i niech $(G_t)_{t\in AT}$, gdzie $AT=\{a,b,c,d\}$, będzie ustalonym namiarem.

Załóżmy dodatkowo, że

$V_c\cap W_c\neq\emptyset$, $V_d\cap W_d\neq\emptyset$,

$V_c\cap W_c\subset G_c$, $V_d\cap W_d\subset G_d$,

$G_a\subset V_a$, $G_a\subset W_a$, $G_b\subset V_b$, $G_b\subset W_b$,

$V_c\not\subset G_c$, $W_c\not\subset G_c$, $V_d\not\subset G_d$, $W_d\not\subset G_d$.

Biorąc pod uwagę pierwszą wersję problemu namiarowania dla danego SRW i danego namiaru stwierdzamy, że odpowiedni warunek nie jest spełniony. Odpowiedź będzie jednak pozytywna, jeśli weźmiemy pod uwagę informację daną implicite postaci

$$f_z(t)=V_t\cap W_t \quad \text{dla } t\in AT.$$

Jednak w przypadku większej liczby obiektów i atrybutów byłoby rzeczą bardzo trudną generować i analizować wszystkie nietrywialne informacje.

Niech $M=(AT,(VAL_t)_{t\in AT},m)$ będzie modelem dla **S**. Na podstawie danych zależności można skonstruować zbiór formul Φ zawierający następujące formuły:

$$(x_a,G_a)\to(x_a,V_a) \tag{1}$$

$$(x_a,G_a)\to(x_a,W_a) \tag{2}$$

$$(x_b,G_b)\to(x_b,V_b) \tag{3}$$

$$(x_b,G_b)\to(x_b,W_b) \tag{4}$$

$$(x_c,V_c\cap W_c)\to(x_c,G_c) \tag{5}$$

$$(x_d,V_d\cap W_d)\to(x_d,G_d) \tag{6}$$

$$((x_a,V_a)\wedge(x_b,V_b))\to((x_c,V_c)\wedge(x_d,V_d)) \tag{7}$$

$$((x_a,W_a)\wedge(x_b,W_b))\to((x_c,W_c)\wedge(x_d,W_d)) \tag{8}$$

Podzbiór Φ zawierający formuły (1)-(6) jest oczywiście prawdziwy w **M**, a elementy tego zbioru opisują inkluzje między odpowiednimi zbiorami. Formuły (7), (8) opisują fakty zawarte explicite w SRW.

Udowodnimy teraz, że następująca formula γ:

$$((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_c, G_c) \wedge (x_d, G_d))$$

reprezentująca namiar jest wnioskowalna z Φ.

Z (1), (3) i (2), (4) na podstawie 3.2(a) otrzymujemy:

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_a, V_a) \wedge (x_b, V_b)) \quad (9)$$

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_a, W_a) \wedge (x_b, W_b)) \quad (10)$$

Z (9), (7) i (10), (8) na podstawie 3.2(c) mamy:

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_c, V_c) \wedge (x_d, V_d)) \quad (11)$$

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_c, W_c) \wedge (x_d, W_d)) \quad (12)$$

Stosując 3.2 (b) do (11), (12) dostajemy

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_c, V_c) \wedge (x_c, W_c) \wedge (x_d, V_d) \wedge (x_d, W_d))$$

i używając aksjomatu A6

$$\Phi \vdash ((x_a, G_a) \wedge (x_b, G_b)) \rightarrow ((x_c, V_c \cap W_c) \wedge (x_d, V_d \cap W_d)). \quad (13)$$

Z (5), (6) na podstawie 3.2(a) otrzymujemy:

$$\Phi \vdash ((x_c, V_c \cap W_c) \wedge (x_d, V_d \cap W_d)) \rightarrow ((x_c, G_c) \wedge (x_d, G_d)), \quad (14)$$

a z (13), (14) na podstawie 3.2(c) dostajemy $\Phi \vdash \gamma$. Zauważmy, że w ten sposób można wywnioskować dowolną informację daną implicite przez SRW.

Z powyższych rozważań wynika, że aby utworzyć zbiór M-formuł wyrażających zdobytą na podstawie doświadczeń wiedzę, a następnie zastosować mechanizm wnioskowania dla danego namiaru, wystarczy zbudować zbiór bardzo prostych formuł implikacyjnych odpowiadających inkluzjom pomiędzy odpowiednimi zbiorami wartości atrybutów. Komputerowa realizacja procesu wnioskowania mogłaby wykorzystywać mechanizmy języka Prolog.

Zaproponowana metoda pozwala na rozwiązywanie pierwszej wersji problemu namiarowania. W przypadku pozostałych nieodzowne wydają się metody stosowane w analizie tablic decyzyjnych.

4. Wnioski

Specyfika wielu zjawisk nie pozwala na skonstruowanie SRW jako zbioru wszystkich wyobrażalnych faktów. Jest więc rzeczą naturalną poszukiwanie możliwości zastosowania aparatu dedukcji do zbioru odnotowanych zdarzeń. Przy takim podejściu proces wnioskowania sprowadza się do poszukiwania dowodu twierdzenia odpowiadającego pytaniu kierowanemu pod adresem bazy wiedzy. Zaproponowana metoda spełnia ten postulat. Warto zauważyć, że jest ona na tyle ogólna, że może być stosowana nie tylko do SRW zawierających informacje w postaci formuł implikacyjnych.

Z punktu widzenia logiki, proponowany system może być zredukowany do teorii otwartej zawierającej jedynie predykaty jednoargumentowe. Z tego samego punktu widzenia jest rzeczą nieistotną, że system jest wielosortowy. Celem było przede wszystkim skonstruowanie aparatu wnioskowania opartego wprost na strukturze SRW. Proponowane podejście nawiązuje do prac wielu badaczy zajmujących się tzw. dynamiką wiedzy i postulujących traktowanie wiedzy jako zbioru formuł pewnej logiki.

Dodatek: Podstawy systemów informacyjnych

1. Systemy informacyjne - definicja i podstawowe własności

System informacyjny jest czwórką:
$S = (X, A, V, f)$
gdzie:
X - skończony zbiór obiektów; A - skończony zbiór atrybutów;
$V = \bigcup_{a \in AT} V_a$ - zbiór wartości atrybutów, przy czym V_a jest zbiorem wartości atrybutu a;
f: $X \times A \to V$ jest funkcją całkowitą taką, że $f(x,a) \in V_a$ dla każdego $x \in X$, $a \in A$.
Przykładem może być system, którego obiektami są ludzie, atrybutami nazwisko, imię, rok urodzenia, zawód itp. Można go więc utożsamiać z relacyjną bazą danych lub tabelką, której wiersze mianowane są obiektami, kolumny zaś atrybutami.

Informacja o obiekcie x jest funkcją $f_x: A \to V$ taką, że $f_x(a) = f(x,a)$ dla każdego $a \in A$. Informację można więc utożsamiać z wierszem tabelki przedstawiającej system informacyjny.

Obiekty $x, y \in X$ są nieodróżnialne w S ze względu na atrybut $a \in A$ (symbolicznie $x \underset{a}{\sim} y$) wtedy i tylko wtedy, gdy $f_x(a) = f_y(a)$

Obiekty $x, y \in X$ są nieodróżnialne w S ze względu na zbiór atrybutów $A' \subset A$ wtedy i tylko wtedy, gdy $f_x(a) = f_y(a)$ dla każdego $a \in A'$.

Obiekty $x, y \in X$ są nieodróżnialne w S wtedy i tylko wtedy, gdy $f_x(a) = f_y(a)$ dla każdego $a \in A$.

W każdym systemie S relacje $\underset{a}{\sim}$ i $\underset{S}{\sim}$ są relacjami równoważności określonymi na zbiorze obiektów X i spełniają warunek $\underset{S}{\sim} = \bigcap_{a \in A} \underset{a}{\sim}$

Każda relacja równoważności dzieli zbiór, na którym jest określona, na niepuste, rozłączne klasy abstrakcji, które nazywamy blokami (lub zbiorami) elementarnymi systemu S. Na przykład atrybut ROK URODZENIA dzieli obiekty systemu na bloki, z których każdy zawiera osobników urodzonych w tym samym roku. Atrybut KOLOR OCZU dzieli obiekty systemu na bloki zawierające osobników o jednakowym kolorze oczu. Iloczyn tych podziałów dzieli zbiór obiektów systemu na bloki, w których znajdują się obiekty o jednakowym roku urodzenia i kolorze oczu.

Jeśli system S dany jest tabelką tab.1, to podział na bloki jest następujący:
- atrybut a dzieli zbiór X na bloki
  $B_1=\{x_1, x_2, x_3\}$, $B_2=\{x_4, x_5, x_6\}$;
- atrybut b dzieli zbiór X na bloki
  $B_3=\{x_1, x_2, x_4, x_5\}$, $B_4=\{x_3, x_6\}$;
- podział odpowiadający iloczynowi podziałów składa się z bloków
  $B_5=\{x_1, x_2,\}$, $B_6=\{x_4, x_5,\}$, $B_7=\{x_3,\}$, $B_8=\{x_6\}$.

Systemy S i S' są równoważne wtedy i tylko wtedy, gdy mają ten sam zbiór obiektów i generują tę samą relację nieodróżnialności na zbiorze X.

Niech S będzie systemem informacyjnym, E_s niech oznacza zbiór wszystkich bloków elementarnych systemu S. Niech $f^*: E_S \to V$ będzie zdefiniowana następująco:

$f^*(e,a) = v$, $e \in E_S$, wtedy i tylko wtedy, gdy istnieje takie $x \in e$, że $f(x,a) = v$.

Jeśli $S = (X, A, V, f)$ jest systemem informacyjnym, to $S^* = (X, A, V, f^*)$ nazywamy jego reprezentacją.

Na przykład reprezentacją systemu S danego tabelką

X	a	b
x_1	p_1	q_1
x_2	p_1	q_1
x_3	p_1	q_2
x_4	p_2	q_1
x_5	p_2	q_1
x_6	p_2	q_2

tab. 1

jest system

X	a	b
$\{x_1, x_2\}$	p_1	p_1
$\{x_3\}$	p_1	q_2
$\{x_4, x_5\}$	p_2	q_1
$\{x_6\}$	p_2	q_2

tab. 2

W systemie będącym reprezentacją każda informacja występuje tylko raz, a obiektami są bloki systemu S.

Często zdarza się, że wartość jakiegoś atrybutu zależy od wartości innych atrybutów w systemie, tzn. jest ona jednoznacznie wyznaczona przez wartości innych atrybutów. Np. adres pacjenta automatycznie wyznacza adres jego przychodni rejonowej.

Niech a, b będą dwoma atrybutami w systemie S. Atrybut b zależy od atrybutu a, (symbolicznie $a \to b$) gdy $\underset{a}{\sim} \subset \underset{b}{\sim}$. Atrybuty a, b są niezależne w S, gdy a nie zależy od b i b nie zależy od a. Atrybuty są równoważne w S (symbolicznie $a \sim b$), gdy $\underset{a}{\sim} = \underset{b}{\sim}$.

Łatwo zauważyć, że jeśli $a \to b$, to istnieje funkcja $g_a^b: V_a \to V_b$ taka, że $g_a^b(f_x(a)) = f_x(b)$, stąd zależności atrybutów zwane są zależnościami funkcjonalnymi. W powyższym przykładzie (tab. 1), atrybuty są wzajemnie niezależne.

Definicję zależności atrybutów można w naturalny sposób rozszerzyć na przypadek, gdy atrybut a zależy od podzbioru B zbioru atrybutów A. W ogólnym przypadku można mówić o zależności zbioru atrybutów C od zbioru atrybutów B (symbolicznie $B \to C$), gdzie B, C są podzbiorami zbioru atrybutów A w systemie S. Tak więc $B \to C$ wtedy i tylko wtedy, gdy $\underset{B}{\sim} \subset \underset{C}{\sim}$. Ponieważ $B \to C$ w S implikuje $B \to C$ w reprezentacji S^* systemu S, więc sprawdzanie zależności atrybutów można przeprowadzić w reprezentacji systemu, co często jest znacznie wygodniejsze. Jeśli

atrybut b zależy w systemie S od atrybutu a, to b jest zbędny w S, o ile oczywiście znana jest funkcja określająca tę zależność. System wyjściowy i system bez atrybutu b dają ten sam podział zbioru X, stąd zgodnie z definicją są to systemy równoważne. Oczywiście mogą istnieć powody, dla których usunięcie zależnych atrybutów z systemu nie jest wskazane, mimo że z punktu widzenia podziału zbioru X na bloki są one zbędne.

Zbiór atrybutów jest minimalny, gdy usunięcie choćby jednego z nich wpływa na podział na bloki elementarne.

Zbiór atrybutów $B \subset A$ nazywamy reduktem zbioru A wtedy i tylko wtedy, gdy $\underset{B}{\sim} = \underset{A}{\sim}$ oraz nie istnieje właściwy podzbiór $B' \subset B$ taki, że $\underset{B'}{\sim} = \underset{A}{\sim}$.

System ze zredukowanym zbiorem atrybutów nazywamy systemem zredukowanym.

Możliwość zredukowania systemu ma często duże znaczenie praktyczne. W przypadku stawiania diagnozy medycznej może się np. okazać, że dla uzyskania takich samych rezultatów wystarcza mniejszy zbiór atrybutów. W takim przypadku bierze się pod uwagę tylko te symptomy, które są istotne dla postawienia prawidłowej diagnozy. Trwają badania mające na celu stworzenie efektywnych algorytmów wyznaczania reduktów.

2. Języki systemów informacyjnych

Języki systemów informacyjnych są typowymi językami zapytań. Pytania dzielą się na:

- mnogościowe („podaj wszystkie książki ..."); odpowiedź - zbiór obiektów,
- relacyjne („podaj wszystkich przodków ..."); odpowiedź - zbiór obiektów lub zbiór par obiektów,
- liczbowe („ile jest osób znających język ..."); odpowiedź - liczba,
- numeryczne („podaj średnie wynagrodzenie ..."); odpowiedź - liczba,
- logiczne („czy X ma oczy niebieskie"); odpowiedź - TAK lub NIE.

Język może obejmować jeden lub więcej rodzajów pytań.

Termy typowego języka:

1) stałe 0 i 1;
2) deskryptory systemu S, czyli pary stałych (a, v), $a \in A$, $v \in V_a$;
3) jeśli t, r są termami, to są nimi również $\neg t$, $t + r$, $t \cdot r$, $t \rightarrow r$, $t \leftrightarrow r$.

Semantyka jest określona w naturalny sposób, np. znaczeniem termu (i odpowiedzią na odpowiednie pytanie) $\neg$(OCZY, NIEBIESKIE) jest zbiór osób mających oczy o kolorze różnym od niebieskiego.

Term t jest prawdziwy w systemie S, gdy $\sigma(t) = X$, gdzie σ jest semantyką języka. Jeśli term jest prawdziwy w S, to nie oznacza to oczywiście, że jest prawdziwy w każdym systemie. Istnieją jednak termy prawdziwe w każdym systemie, np. $\neg t + t$.

Zbiór $Y \subset X$ jest zbiorem opisywalnym w S wtedy i tylko wtedy, gdy istnieje taki term t, że $\sigma(t) = Y$.

W zależności od konstrukcji języka, można ustalić reguły przekształcania termów, zdefiniować ich postać normalną, badać dokładność i efektywność języka. Zbiory elementarne systemu informacyjnego odgrywają bardzo ważną rolę w obliczaniu wartości termów, a do tego sprowadza się w istocie zagadnienie poszukiwania odpowiedzi na pytania formułowane w stosunku do systemu (w komputerowej realizacji

- do bazy danych). Na przykład, jeśli t jest termem prostym, tzn. termem postaci $(a_1, v_1)\cdot(a_2, v_2)\cdot\ldots\cdot(a_n, v_n)$, to $\sigma_S(t)$ jest zbiorem elementarnym w systemie S lub zbiorem pustym. Wiele prac poświęcono zagadnieniu zastosowania pojęcia zbioru elementarnego do badań dokładności i efektywności języków oraz do konstruowania efektywnych algorytmów obliczania wartości termów.

Oprócz badania własności „klasycznych" systemów informacyjnych i ich języków, wielu autorów zajmuje się ich licznymi odmianami: systemami wielostopniowymi, hierarchicznymi, rozproszonymi, wielowartościowymi, stochastycznymi i przybliżonymi. Systemy wielowartościowe i przybliżone charakteryzują się tym, że wartości atrybutów w tych systemach są zbiorami. Interpretacja tych zbiorów w każdym z tych systemów jest jednak zupełnie inna. W przypadku systemów wielowartościowych, jeśli wartością atrybutu JĘZYK jest {ANGIELSKI, NIEMIECKI, CHIŃSKI}, to oznacza, że dana osoba zna dokładnie te trzy języki. W systemie przybliżonym, w identycznym przypadku wartość atrybutu interpretuje się w ten sposób, że dana osoba zna na pewno któryś z tych trzech języków, nie jesteśmy jednak z jakichś powodów określić, który z nich. Ciekawe efekty uzyskano, stosując do opisu takich systemów języki logik modalnych.

3. Zagadnienia klasyfikacji obiektów

Przedstawione powyżej zagadnienia dają się w istocie sprowadzić do następującego problemu: dany jest system informacyjny $S = (X, A, V, f)$, jego język L_S oraz term t należący do języka L_S; należy znaleźć zbiór $Y \subset X$ opisany przez ten term. Problem ten jest podstawą systemów wyszukiwania informacji.

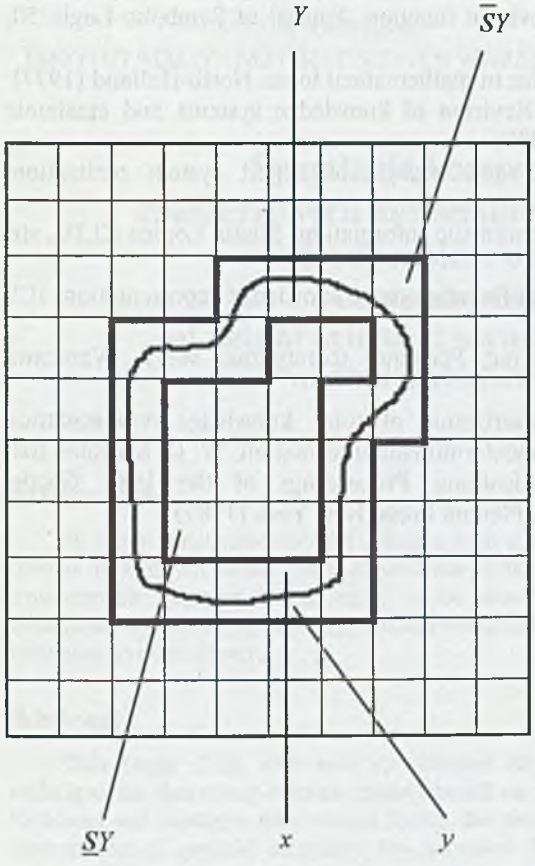
Zajmiemy się teraz następującym zagadnieniem, będącym podstawą tzw. sztucznej inteligencji: dany jest system informacyjny $S = (X, A, V, f)$, jego język L_S oraz pewien zbiór $Y \subset X$; należy znaleźć term t języka L_S taki, że t jest opisem przybliżonym zbioru Y. Wyjaśnimy to na przykładzie.

Lekarz specjalista przegląda zdjęcia rentgenowskie płuc pacjentów i na podstawie swej wiedzy i doświadczenia klasyfikuje pacjentów na zdrowych i chorych. Ekspert nie musi oczywiście uzasadniać swojej decyzji. Gdyby jednak chciał on kogoś nauczyć rozpoznawać osoby chore na podstawie zdjęć, musiałby umieć scharakteryzować swoje decyzje podając, że zdjęcia osób chorych mają określone cechy świadczące o istnieniu jednostki chorobowej. Gdyby taki opis udało się wykonać, to dowolna osoba, nie mająca wiedzy eksperta i jego doświadczenia, potrafiłaby trafnie dokonać klasyfikacji. Podobny charakter ma nie tylko analiza zdjęć, ale dowolna diagnoza medyczna. Analogiczne problemy występują w wielu innych dziedzinach. Ich istotą jest opisanie zadanego zbioru w odpowiednim języku formalnym. Zagadnienie to jest ściśle związane z tzw. logiką indukcji. Głównym celem tej logiki jest analiza stawiania hipotez na podstawie przykładów. Podany wyżej przykład jest typowym rozumowaniem indukcyjnym; w podanych bowiem przypadkach osób chorych (przypadki rozpoznawane przez eksperta) pytamy, jakie są charakterystyczne symptomy rozpoznawanej choroby.

Niech $S = (X, A, V, f)$ i niech $Y \subset X$. Największy (w sensie inkluzji) zbiór opisywalny w systemie S zawarty w Y nazywamy dolnym przybliżeniem zbioru Y w systemie S i oznaczamy $\underline{SY}$.

Najmniejszy (w sensie inkluzji) zbiór opisywalny w systemie S zawierający zbiór Y nazywamy górnym przybliżeniem zbioru Y w systemie S i oznaczamy $\overline{S}Y$.

Na poniższym rysunku przedstawiono górne i dolne przybliżenie zbioru Y. Każda kratka jest zbiorem elementarnym. Obiekty x, y są nieodróżnialne w S.



Jeśli więc Y jest zbiorem osób chorych wyznaczonych przez eksperta na podstawie jego wiedzy i doświadczenia i chcemy ten zbiór zdefiniować określając wartości odpowiednich atrybutów w systemie $S=(X,A,V,f)$, $Y \subset X$, to z rozważań dotyczących systemów informacyjnych wynika, że zadanie to może być niewykonalne, gdyż w ten sposób można zdefiniować tylko takie podzbiory zbioru X, które są sumami zbiorów elementarnych w systemie S, a więc na podstawie tylko analizy symptomów choroby nie zawsze można odtworzyć diagnozę eksperta. Można ją jedynie odtworzyć z pewnym przybliżeniem. Oczywiście im „drobniejszy" podział na zbiory elementarne (idealna sytuacja, gdy każdy z nich jest zbiorem jednoelementowym, a więc gdy system jest idealnie selektywny), tym opis jest dokładniejszy.

Przedstawiony problem jest punktem wyjścia do wielowątkowych rozważań takich aspektów sztucznej inteligencji jak niepełna klasyfikacja obiektów, indukcja, procesy uczenia się, logiki systemów informacyjnych itp.

Literatura

[1] A. Abramowicz: Bearings Problem in Nondeterministic Information Systems. Bull. of the PAS, Tech. Sciences, Vol. 36, No. 3-4, str. 241-250 (1988)

[2] C. Alchourron, P. Gardenfors, D. Makinson: On the logic of theory change: Partial meet contraction and revision function. Journal of Symbolic Logic 50, str. 510-530 (1985)

[3] J. L. Bell, M. Machover: A course in mathematical logic. North-Holland (1977)

[4] P. Gardenfors, D. Makinson: Revision of knowledge systems and epistemic entrenchment. (w rękopisie) (1988)

[5] A. Mrózek: Rough sets and some aspects of expert system realization. (w rękopisie) (1988)

[6] E. Orlowska: Logic of nondeterministic information. Studia Logica XLIV, str. 93-102 (1985)

[7] E. Orlowska, Z. Pawlak: Logical foundations of knowledge representation. ICS PAS Reports 537 (1984)

[8] Z. Pawlak: Systemy informacyjne. Podstawy teoretyczne. WNT, Warszawa (1983)

[9] D. Vakarelov: Abstract characterization of some knowledge representation systems and the logic NIL of nondeterministic information. W: D. Skorolev (ed) Mathematical logic and applications. Proceedings of the 1986 Goedel Conference, Druzhba, Bulgaria, Plenum Press, New York (1987)

MAREK KACPRZAK
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Symulacja komputerowa sieci rozszerzonych automatów asynchronicznych w środowisku rozproszonym

Computer simulation of nets of extended asynchronous automata in distributed environment

Streszczenie

W pracy przedstawiono sieci rozszerzonych automatów asynchronicznych - formalny opis modeli dyskretnych, wykorzystujący podzbiór języka Estelle (ISO 9074). Omówiono problemy i rozwiązania symulacji równoległej takich sieci w środowisku rozproszonym, złożonym z komputerów osobistych. Celem symulacji równoległej jest zmniejszenie czasów trwania eksperymentów symulacyjnych.

Abstract

This paper deals with nets of extended asynchronous automata. It uses the formal technique for describing discrete models based on a subset of Estelle language (ISO 9074). Problems and solutions encountered during the parallel simulation process in the distributed environment of personal computers are discussed. The goal of this parallel simulation is the reduction of time spent on simulation experiment overall.

1. Symulacja komputerowa

Symulacja komputerowa jest powszechnie stosowaną metodą badawczą, wspomagającą analizę i projektowanie systemów w wielu dziedzinach nauki i techniki.

Metodologia symulacji komputerowej obejmuje następujące etapy:

- określenie zadania symulacji dla rozważanego systemu fizycznego,
- sformułowanie modelu systemu,
- zaprogramowanie modelu - utworzenie programu symulacyjnego,
- sprawdzenie poprawności modelu,
- zaplanowanie eksperymentów symulacyjnych,
- wykonanie eksperymentów symulacyjnych,
- analiza uzyskanych wyników.

Modele opisują systemy przy użyciu pojęć zbioru stanów i zbioru zdarzeń. Zmiany stanu są powodowane przez zdarzenia. W niniejszej pracy są rozpatrywane modele dyskretne, tzn. takie, w których zmiany stanu następują skokowo w ustalonych punktach czasu symulacyjnego. Symulacja prowadzona w oparciu o model dyskretny jest nazywana symulacją dyskretną.

Najczęściej używanym sposobem formalnego opisu modeli dyskretnych są opisy sieciowe, np. numeryczne sieci Petriego z czasem, sieci ewaluacyjne i sieci kolejkowe [1]. W niniejszej pracy przedstawiony jest inny opis sieciowy - sieci rozszerzonych automatów asynchronicznych, omówiony w p.2.

Do tworzenia programów symulacyjnych mogą być używane specjalizowane języki symulacyjne (np. CSL, Simscript, Simula itp.) lub (rzadziej) języki wysokiego poziomu ogólnego przeznaczenia (np. Fortran, Pascal itp.). Obecnie coraz powszechniej widać podejście, które umożliwia pominięcie trudnego i żmudnego samodzielnego tworzenia programu symulacyjnego. Przykładem mogą być pakiety do symulacji i analiz sieci kolejkowych [2], [3]. Po wprowadzeniu do komputera modelu kolejkowego jest tworzony automatycznie program symulacyjny. Przy pewnych założeniach odnośnie sieci kolejkowych, pakiety te umożliwiają także uzyskiwanie rozwiązań analitycznych.

W przypadku sieci rozszerzonych automatów asynchronicznych, istnieją sposoby automatycznego przejście od formalnego opisu sieci automatów do pakietów symulacji i analiz sieci kolejkowych [4] lub do programu symulacyjnego w języku programowania obiektowego [5].

Utworzenie programu symulacyjnego umożliwia, na podstawie próbnych eksperymentów symulacyjnych, sprawdzenie poprawności przyjętego modelu. Jeżeli wyniki symulacji odbiegają od obserwacji istniejącego systemu lub różnią się od wyników oczekiwanych w przypadku projektowanego systemu, konieczna jest zmiana modelu. Jeżeli wyniki próbnego eksperymentu symulacyjnego są akceptowane, wówczas wykonywane są właściwe eksperymenty symulacyjne, zgodnie z ustalonym planem.

Uzyskane wyniki symulacji są poddawane analizie i interpretacji. Zazwyczaj, wobec losowego charakteru modelowanego systemu, wyniki muszą być poddane obróbce statystycznej.

Symulacja dyskretna może być wykonywana na komputerze sekwencyjnym (symulacja sekwencyjna) lub równolegle na połączonych zbiorach procesorów (symulacja równoległa). Zbiór procesorów ze wspólną pamięcią lub/i wspólnym zegarem będziemy nazywać komputerem równoległym. Zbiór procesorów nie posiadających wspólnej pamięci lub wspólnego zegara, połączonych ośrodkiem komunikacyjnym, będziemy nazywać środowiskiem rozproszonym.

2. Sieci rozszerzonych automatów asynchronicznych

Rozszerzony automat jest fundamentalnym pojęciem języka Estelle. Język Estelle, opisany w normie międzynarodowej ISO 9074 [6], jest językiem specyfikacji systemów rozproszonych, przeznaczonym głównie do opisu protokołów komunikacyjnych sieci otwartych ISO-OSI [7]. Estelle można traktować jako język Pascal ISO 7185 [8] z pewnymi ograniczeniami i nowymi konstrukcjami, umożliwiającymi, między innymi, zapis równoległości i niedeterminizmu.

Specyfikacja rozpatrywanego systemu w języku Estelle jest dokonywana za pomocą sieci niedeterministycznych, rozszerzonych automatów ze skończoną liczbą stanów

sterowania. Stan rozszerzonego automatu jest opisywany przez bieżący stan sterowania oraz dodatkowe zmienne. Dla każdego automatu są określone podzbiory początkowych stanów sterowania; wybór jednego z nich jest niedeterministyczny. Zmiana stanu automatu jest nazywana tranzycją. Działanie automatu odbywa się w dwóch fazach: wyboru tranzycji do wykonania i wykonania wybranej tranzycji. Jeżeli zbiór tranzycji do wykonania zawiera więcej niż jeden element, to wybór wykonywanej tranzycji jest niedeterministyczny. W pracach [9], [5] opisano sposób wprowadzenia do specyfikacji w języku Estelle czasów wykonania faz wyboru tranzycji i wykonania tranzycji. Czasy te mogą być zmiennymi losowymi, stanowiąc dodatkowy „wymiar" niedeterminizmu języka Estelle.

Automaty mogą współpracować ze sobą wysyłając przez punkty interakcji (porty) wiadomości zwane interakcjami lub odbierając interakcje przysłane do punktów interakcji. Interakcje mogą zawierać parametry. Odbierane interakcje są gromadzone w kolejkach z regulaminem naturalnym (FIFO). Z każdym punktem interakcji automatu może być związana odrębna kolejka lub kilka punktów interakcji automatu może mieć wspólną kolejkę.

Wykonanie tranzycji jest opisywane jako treść procedury w sensie języka Pascal. Procedura taka (zwana dalej procedurą tranzycji) może zawierać większość instrukcji języka Pascal oraz nowe instrukcje języka Estelle:

- powołania lub zniszczenia automatu-dziecka,
- połączenia lub rozłączenia punktów interakcji (przesyłanie interakcji przez punkty interakcji jest możliwe tylko wtedy, gdy punkty interakcji zostały wcześniej połączone),
- wysłania interakcji,
- pewne instrukcje niedeterministyczne.

Automaty mogą powoływać automaty-dzieci. Zależnie od atrybutów automatu-ojca, wszystkie automaty-dzieci mogą wykonywać tranzycje równolegle lub tylko jeden automat-dziecko (wybrany niedeterministycznie) może wykonywać tranzycje. Współpraca automatów-ojców z automatami-dziećmi może odbywać się nie tylko przez wysyłanie interakcji, ale także przez wspólne zmienne.

Możliwość dynamicznego powoływania automatów-dzieci i specyficzne zasady wykonywania tranzycji przez automaty-dzieci są istotne przy specyfikacji protokołów komunikacyjnych sieci komputerowych, w których automaty-dzieci reprezentują najczęściej połączenia sieciowe. Systemy rozproszone różne od sieci komputerowych, stanowiące główny obszar zainteresowań autora, np. komputerowo integrowanego wytwarzania (CIM), ekonomiczne czy rozproszone bazy danych, mogą być specyfikowane w języku Estelle bez powoływania automatów-dzieci. W związku z tym, w niniejszej pracy wprowadzono istotne ograniczenie na język Estelle: automaty nie mogą powoływać automatów-dzieci. Przyjęcie tego założenia oznacza, że:

- instrukcja wysłania interakcji jest jedyną instrukcją języka Estelle, która może występować w procedurze tranzycji,
- przesyłanie interakcji jest jedynym sposobem współpracy automatów,
- współpraca działających równolegle automatów ma wyłącznie charakter asynchroniczny.

Formalnym językiem opisu sieci rozszerzonych automatów asynchronicznych jest więc podzbiór języka Estelle. Tylko proste protokoły komunikacyjne mogą być opisane przez sieci rozszerzonych automatów asynchronicznych.

3. Symulacja sekwencyjna sieci rozszerzonych automatów asynchronicznych

Cechą charakterystyczną rozwiązywania realistycznych problemów przy użyciu symulacji sekwencyjnej jest duża złożoność czasowa symulacji - tzn. długie czasy trwania eksperymentów symulacyjnych.

Złożoność czasowa symulacji wynika z różnych powodów.

* Symulowane modele zawierają zazwyczaj wielowymiarową przestrzeń parametrów, z czego wynika obszerny program badań.

* W trakcie symulacji wyliczanych jest wiele wskaźników i statystyk. Standardowymi statystykami są: wartości minimalna, maksymalna, średnia i wariancja. Często mogą być konieczne wyliczenia także momentów wyższych rzędów.

* Wyniki symulacji mają charakter losowy. Wynika on nie tylko z niedeterminizmu automatów, opisanego w p.2, ale także ze specyfiki symulacji sieci automatów: np. jeżeli dwa automaty w tej samej chwili wysyłają interakcje do wspólnej kolejki trzeciego automatu, to kolejność zapisania interakcji do kolejki jest przypadkowa, co wpływa na wyniki liczbowe symulacji. Podanie samej średniej i wariancji nie wystarcza - konieczne jest jeszcze określenie przedziałów ufności. Dla zawężenia tych przedziałów każdy eksperyment musi być powtórzony wielokrotnie (metoda wielokrotnych przebiegów) lub musi być prowadzony odpowiednio długo (metoda jednego przebiegu) [10]. Obie metody mogą być stosowane w sytuacji, gdy symulowany system osiąga stan stacjonarny, ale jeżeli osiągana jest tylko faza przejściowa, to konieczne jest stosowanie metody wielokrotnych przebiegów.

Złożoność czasową symulacji potwierdzają opisane w pracy [5] wyniki symulacji sieci rozszerzonych automatów asynchronicznych, modelujących szybką komputerową sieć lokalną. Badania dotyczyły przepustowości implementacji prostego protokołu komunikacyjnego. Eksperymenty symulacyjne prowadzono na komputerze osobistym z mikroprocesorem Intel 80486, z zegarem 50 MHz. Uproszczony model zawierał 5 parametrów, co oznaczało konieczność przeprowadzenia badań dla około tysiąca zestawów parametrów. Głównym celem było badanie przepustowości, badano także „wąskie gardła" w wymianie informacji, tzn. statystyki odnośnie wszystkich kolejek interakcji (długość kolejki, czasy oczekiwania), wszystkich automatów (czas przejścia wiadomości przez automaty) i wybranych zasobów - np. buforów na dane (stopień wykorzystania). Zastosowano metodę wielokrotnych przebiegów symulacyjnych, powtarzając każdy eksperyment pięciokrotnie, co oznaczało konieczność wykonania około pięciu tysięcy eksperymentów (każdy o czasie trwania około 2 minut), wymagających łącznie blisko 7 dni (!) ciągłej pracy komputera.

Jedyną szansą skrócenia czasów symulacji sekwencyjnej jest stosowanie jak najszybszych komputerów. Jednakże, ze względów technologicznych, możliwości zwiększania szybkości działania komputerów sekwencyjnych są ograniczone. Tak więc w przypadku symulacji bardziej skomplikowanego systemu złożoność czasowa symulacji może uniemożliwiać prowadzenie eksperymentów symulacyjnych w akceptowalnym czasie.

W tej sytuacji, podstawowym sposobem zmniejszenia czasów trwania eksperymentów symulacyjnych jest zastosowanie symulacji równoległej. Szczególnie

interesująca ze względów praktycznych jest symulacja w środowisku rozproszonym, która może być wykonywana np. na połączonych komputerach osobistych (np. klasy IBM PC), bez potrzeby korzystania z na ogół trudno dostępnych i drogich komputerów równoległych.

4. Stan prac nad symulacją równoległą

Symulacja równoległa jest przedmiotem intensywnych badań od blisko 15 lat [11], [12]. Algorytmy symulacji równoległej są odpowiednimi modyfikacjami algorytmów symulacji sekwencyjnej.

4.1. Algorytmy symulacji sekwencyjnej

Ze względu na sposób realizacji upływu czasu symulacyjnego, stosowane algorytmy dzielą się na dwie typy: ze stałym krokiem i ze zmiennym krokiem. W algorytmie stałego kroku wybór wielkości kroku upływu czasu symulacyjnego decyduje o dokładności modelu. Im mniejszy krok, tym dokładniejszy model, ale równocześnie spada szybkość symulacji. Algorytm jest stosowany głównie wtedy, gdy zdarzenia występują w równych odstępach czasu. W innych przypadkach stosowany jest algorytm zmiennego kroku.

Ze względu na sposób sterowania przebiegiem symulacji, opisywane w literaturze algorytmy dzielą się na trzy typy: planowania zdarzeń, przeglądania działań i interakcji procesów [1].

Z punktu widzenia symulacji równoległej najważniejszy jest algorytm planowania zdarzeń ze zmiennym krokiem symulacji. W algorytmie tym wykorzystywane są trzy struktury danych - lista zdarzeń, stan systemu i czas systemowy. Lista zdarzeń zawiera zaplanowane zdarzenia wraz z etykietą czasową (ang. timestamp), określającą w której chwili przyszłego czasu symulacyjnego dane zdarzenie wystąpi. Program sterowania symulacją:

- wybiera i usuwa z listy zdarzenie o najmniejszej wartości etykiety,
- powołuje program obsługi tego zdarzenia (który może zmienić stan systemu),
- planuje następne zdarzenie i wraz z wyliczoną etykietą zapisuje je do listy zdarzeń,
- nadaje zmiennej reprezentującej czas systemowy nową wartość, równą etykiecie obsłużonego zdarzenia.

Symulacja kończy się po wyczerpaniu zdarzeń do obsługi lub po spełnieniu jakiegoś warunku, np. osiągnięciu horyzontu czasowego symulacji.

4.2. Źródła równoległości w symulacji sekwencyjnej

* Równoległość aplikacji

W przypadku metody wielu przebiegów symulacyjnych, rozwiązaniem najbardziej oczywistym jest wykonywanie tej samej symulacji, ale dla różnych zestawów danych, równolegle w czasie, na wielu niezależnych procesorach. Takie rozwiązanie nie może być stosowane w przypadku ograniczenia wielkości pamięci niezbędnej do symulacji. Ten typ równoległości może być również zrealizowany na komputerze równoległym o architekturze SIMD (synchroniczna praca procesorów ze wspólnym strumieniem instrukcji z wieloma strumieniami danych).

* Równoległość procedur

W przypadku metody jednego przebiegu symulacyjnego, skrócenie czasu symulacji jest możliwe dzięki wydzieleniu pewnych procedur, które mogą być wykonywane równolegle na różnych procesorach, np. generatory liczb losowych i programy obróbki danych statystycznych mogą być oddzielone od głównego zadania - symulacji występowania zdarzeń. Konieczne jest zapewnienie wymiany danych między użytymi procesorami.

* Równoległość składowych modelu

Konieczna jest dekompozycja modelu na składowe. Każda składowa jest realizowana na oddzielnym procesorze. Najprostszy sposób dekompozycji polega na uwzględnieniu naturalnej równoległości modelu, często wynikającej bezpośrednio ze sposobu opisu modelu.

W zdekomponowanym modelu może być stosowana scentralizowana lub zdecentralizowana lista zdarzeń. Scentralizowana lista zdarzeń wymaga współpracy procesorów na zasadzie procesor główny (z listą zdarzeń) - procesory podległe (reprezentujące składowe modelu).

Podstawowym zadaniem procesora głównego jest zapewnienie właściwej kolejności zdarzeń w czasie - przestrzegania reguły, że skutek nie może poprzedzać przyczyny (ang. causality rule).

4.3. Zadanie symulacji równoległej

Zadanie symulacji równoległej można sformułować następująco:

opracować zasady symulacji dyskretnej dla jednego eksperymentu symulacyjnego na komputerze równoległym lub w środowisku rozproszonym.

Rozwiązanie tak postawionego zadania umożliwia jedynie równoległość procedur i składowych.

Wyniki symulacji równoległej muszą być, oczywiście, takie same jak w przypadku symulacji sekwencyjnej. Czas wykonania symulacji równoległej powinien być nie dłuższy od czasu wykonania symulacji sekwencyjnej.

Miarą efektywności symulacji równoległej jest przyśpieszenie obliczeń, definiowane dla danego problemu jako stosunek minimalnego czasu wykonania symulacji sekwencyjnej do czasu wykonania symulacji równoległej. Miarą stopnia wykorzystania procesorów zastosowanych do symulacji równoległej jest stosunek przyśpieszenia obliczeń do liczby procesorów użytych w celu uzyskania tego przyśpieszenia [13].

4.4. Symulacja procesów logicznych

Prace nad symulacją równoległą dotyczą głównie równoległości składowych modelu ze zdecentralizowaną listą zdarzeń. Najczęściej stosowanym sposobem dekompozycji systemu fizycznego jest dekompozycja na składowe - procesy logiczne. W podejściu tym dokonuje się podziału zbioru stanów na parami rozłączne podzbiory. Każdy podzbiór stanów opisuje jedną składową. Każda składowa jest reprezentowana przez proces logiczny. Procesy logicznie działają równolegle na różnych procesorach. Jedyną formą współdziałania procesów jest przesyłanie komunikatów przez ośrodek komunikacyjny (inne formy współdziałania, np. przez zmienne dzielone nie są

możliwe). Komunikaty służą do przesyłania danych i do synchronizacji. Zakłada się, że ośrodek jest idealny (nie powoduje błędów, utraty, ani powielenia komunikatów). Czas przesłania komunikatów przez ośrodek jest niezerowy. Każdy proces logiczny wykonuje symulację wg. algorytmu planowania zdarzeń. Są dwa rodzaje zdarzeń: wewnętrzne, wpływające tylko na własny stan składowej, oraz zewnętrzne, wpływające na stany innych składowych. Z każdym procesem logicznym związany jest lokalny czas symulacyjny.

Działanie procesów logicznych może mieć charakter synchroniczny lub asynchroniczny.

4.4.1. Synchroniczna symulacja procesów logicznych

W synchronicznej symulacji procesów logicznych wszystkie procesy logiczne w danej chwili czasu rzeczywistego mają identyczną wartość lokalnego czasu symulacyjnego.

Możliwe jest stosowanie algorytmu upływu czasu symulacyjnego o stały lub o zmienny krok (w tym przypadku konieczny jest protokół uzgadniania maksymalnej wielkości kroku). Globalny zegar czasu symulacyjnego może być scentralizowany (zegarem steruje wydzielony procesor) lub zdecentralizowany (stan zegara jest uzgadniany między procesami logicznymi).

4.4.2. Asynchroniczna symulacja procesów logicznych

W asynchronicznej symulacji procesów logicznych procesy logiczne w danej chwili czasu rzeczywistego mogą mieć różne wartości lokalnego czasu symulacyjnego. Zasadniczym problemem jest protokół synchronizacji procesów logicznych.

W literaturze za podstawowe, najczęściej opisywane i najlepiej zbadane, uważa się dwa typy protokołów synchronizacji: zachowawcze (ang. conservative) i optymistyczne (ang. optimistic).

W obu typach protokołów komunikaty zawierają informację o zdarzeniu zewnętrznym procesu-nadawcy i etykietę czasową, określającą czas wystąpienia tego zdarzenia, odmierzany według lokalnego zegara procesu-nadawcy.

* Protokoły zachowawcze

W wersji podstawowej protokołu zachowawczego (zwanego też CBM - od nazwisk autorów: Chandy, Misra, Bryant) obowiązuje zasada, że obsługa zdarzenia następuje tylko wtedy, gdy jest pełna gwarancja, że żaden z procesów nie przyśle komunikatu z etykietą o wartości mniejszej od lokalnego zegara procesu-odbiorcy.

Po dekompozycji modelu na składowe następuje statyczne przyporządkowanie każdemu procesowi wszystkich jego procesów-nadawców. Komunikaty przychodzące do procesu-odbiorcy są gromadzone w kolejkach FIFO. Z każdą kolejką jest związana zmienna, reprezentująca najmniejszą wartość etykiety spośród zapisanych w kolejce lub, gdy kolejka jest pusta, ostatnią wartość etykiety. Proces wybiera do obsługi zdarzenie wewnętrzne ze swojej lokalnej listy zdarzeń lub komunikat z kolejki o najmniejszej wartości etykiety zdarzeń zewnętrznych, zachowując właściwą kolejność zdarzeń w czasie. Jeżeli wybrana kolejka jest pusta, to proces musi zostać zablokowany do chwili przyjścia komunikatu do tej kolejki.

Oczekiwanie na komunikaty jest potencjalnym źródłem zakleszczeń, występujących wtedy, gdy kilka procesów oczekuje na komunikat od innego, który nie jest w stanie go wysłać.

Opracowano wiele wersji protokołów zachowawczych, w których stosowane są różne sposoby przeciwdziałania zakleszczeniom (np. przez przesyłanie dodatkowych komunikatów synchronizacyjnych) lub specjalne procedury w przypadku wystąpienia zakleszczeń itp.

* Protokoły optymistyczne

W wersji podstawowej protokołu optymistycznego (zwanego też Time Warp) obowiązuje zasada, że obsługa zdarzenia następuje również wtedy, gdy nie ma gwarancji, że żaden z procesów nie przyśle komunikatu z etykietą o wartości mniejszej od lokalnego zegara procesu-odbiorcy.

Powiązania między procesami-odbiorcami i procesami-nadawcami mogą być dokonywane dynamicznie. Jeżeli któryś z procesów-odbiorców odbierze taki komunikat, to następuje takie cofnięcie symulacji, aby była spełniona zasada właściwej kolejności zdarzeń. Cofnięcie symulacji może wymagać nie tylko przywrócenia wcześniejszego stanu procesu-odbiorcy, ale także anulowania wysłanych już do innych procesów błędnych komunikatów. Anulowanie komunikatów umożliwiają tak zwane komunikaty negatywne.

Opracowano wiele wersji protokołów optymistycznych, w których różnie są rozwiązywane zasady awansowania symulacji w sytuacji, gdy jest prawdopodobna konieczność jej anulowania, oraz reguły cofnięcia symulacji.

* Porównanie protokołów zachowawczych i optymistycznych

Głównymi zaletami protokołów zachowawczych są: małe wymagania na wielkość pamięci, proste struktury danych, prosty algorytm symulacji. Główną zaletą protokołów optymistycznych jest lepsza strategia wykorzystywania równoległości. Jednoznaczne porównanie efektywności obu protokołów, nawet przy identycznej platformie sprzętowo-programowej symulacji równoległej, jest niemożliwe - efektywność zależy od konkretnego symulowanego modelu.

4.3.3. Porównanie synchronicznej i asynchronicznej symulacji procesów logicznych

Porównania obu symulacji zostały dokonane dla szczególnych przypadków. W przypadku wykładniczego rozkładu czasów trwania symulacji w każdym kroku, symulacja asynchroniczna jest co najwyżej log(n) razy szybsza od symulacji synchronicznej (gdzie n jest liczbą wykorzystywanych procesorów). W przypadku równomiernego rozkładu czasów symulacja asynchroniczna trwa 2 razy krócej od synchronicznej (niezależnie od liczby wykorzystywanych procesorów).

5. Problemy naukowo-badawcze w symulacji sieci rozszerzonych automatów asynchronicznych w środowisku rozproszonym

Interesującym nas środowiskiem są komputery osobiste klasy IBM PC, być może różnych typów, z różnymi zegarami, z różnymi systemami operacyjnymi.

Zadanie symulacji równoległej w odniesieniu do tego środowiska formulujemy następująco:

> opracować zasady symulacji dyskretnej dla jednego eksperymentu symulacyjnego, gwarantujące przyśpieszenie obliczeń nie mniejsze od jedności.

Zauważmy, że w pewnych przypadkach uzyskanie przyśpieszenia większego od jedności, nawet przy zerowych czasach przesyłania komunikatów i dowolnej liczbie procesorów nie jest możliwe. Przykładem są dwa automaty: jedyną tranzycją pierwszego jest wysłanie interakcji do drugiego, jedyną tranzycją drugiego jest odebranie interakcji od pierwszego; zadanie symulacji jest ograniczone do wykonywania tranzycji przez oba automaty, żadne inne obliczenia nie są prowadzone. Działanie tych automatów jest sekwencyjne i przyśpieszenie w symulacji równoległej większe od jedności nie jest możliwe.

Sformułowane zadanie wyznacza kilka problemów naukowo-badawczych, z których najważniejsze to: sposób połączenia komputerów używanych do symulacji, wybór algorytmu symulacji, sposób dekompozycji modelu i alokacji procesów, wybór platformy programowej symulacji.

5.1. Sposób połączenia komputerów używanych do symulacji

Warunkiem koniecznym pomyślnego rozwiązania zadania symulacji sieci automatów jest połączenie komputerów tworzących środowisko rozproszone symulacji w taki sposób, aby czas tracony na przesyłanie komunikatów między procesami był jak najkrótszy. Zgrubne oszacowanie pożądanego czasu przesłania jednego bajtu danych między dowolną parą komputerów można sformułować następująco: czas ten powinien być porównywalny z czasem wywołania procedury lokalnej, tzn. być rzędu jednej mikrosekundy. Jest to wymaganie bardzo ostre, w szczególności warunek ten nie może być spełniony w żadnej sieci lokalnej. Realne jest natomiast połączenie komputerów w szybką sieć lokalną ATM lub utworzenie systemu wielokomputerowego metodą sprzęgania magistrali PCI. W obu przypadkach wymagana jest bardzo wysoka jakość transmisji danych.

W przypadku sieci ATM (z transmisją szeregową) komputery powinny być zgrupowane na obszarze o średnicy kilkudziesięciu metrów. Wybór topologii sieci (liczba i rozmieszczenie przełączników) wymaga badań eksperymentalnych. Ze względu na wnoszone narzuty czasowe, niekorzystne będzie korzystanie z jakiegokolwiek standardowego protokołu komunikacyjnego. Właściwym rozwiązaniem jest transmisja datagramowa (bez potwierdzeń), zarówno w warstwie sterowania łączem logicznym, jak i w warstwie transportowej.

W systemie wielokomputerowym stosowana będzie transmisja równoległa. Może być konieczne opracowanie własnych urządzeń sprzęgających magistrale. Wymagane jest takie zgrupowanie komputerów, aby maksymalna odległość między komunikującymi się komputerami wynosiła pojedyncze metry. Wybór topologii (pierścień,

drzewo, hipersześcian itp.) wymaga badań eksperymentalnych. Przesyłanie danych powinno odbywać się ze zminimalizowanym przeplotem sygnałów.

W obu sposobach połączeń komputerów wykrycie przez komputer-odbiorcę błędnych danych powinno doprowadzić do przerwania eksperymentu symulacyjnego, bez możliwości retransmisji błędnych danych i wznowienia symulacji. Kontrolę sekwencyjności danych należy ograniczyć do numerów sekwencyjnych.

5.2. Wybór algorytmu symulacji

Jak wynika z omówienia stanu prac nad algorytmami symulacji równoległej przedstawionego w p.4, podstawowym algorytmem w odniesieniu do sieci automatów jest symulacja procesów logicznych. Naturalnym sposobem dekompozycji sieci automatów jest przyjęcie zasady, że każdy automat jest reprezentowany przez jeden proces, wykonywany na oddzielnym komputerze. Ze względu na założenie, że jedyną formą współdziałania automatów jest przesyłanie interakcji, wymaganie podziału przestrzeni stanów na rozłączne podzbiory jest zawsze spełnione. Zdarzeniami zewnętrznymi, o których jest przekazywana informacja w komunikatach międzyprocesowych, są wysłania przez automaty interakcji do innych automatów.

Zgodnie z p.4, nie można jednoznacznie określić który z algorytmów symulacji procesów logicznych jest najbardziej efektywny. Niewątpliwy wpływ na efektywność algorytmów będzie miała przyjęta platforma sprzętowo-programowa środowiska rozproszonego (np. stosunek szybkości przetwarzania do szybkości przesyłania komunikatów). Dla przykładowych sieci automatów należy więc przeprowadzić testy porównawcze efektywności kilku wybranych wariantów podstawowych algorytmów synchronicznej i asynchronicznej symulacji procesów logicznych. Niewykluczona może być możliwość opracowania własnej wersji algorytmu symulacji, dostosowanego do specyfiki używanego środowiska rozproszonego.

5.3. Sposób dekompozycji modelu i alokacja procesów

Opisana w p.5.2. naturalna dekompozycja modelu sieci automatów i sposób alokacji procesów logicznych nie jest rozwiązaniem jedynym. Podejście takie może być nieoptymalne w sensie minimalizacji czasu trwania symulacji albo wręcz niewykonalne z powodu braku niezbędnej liczby komputerów. Dlatego często właściwym rozwiązaniem może być alokacja więcej niż jednego procesu logicznego, reprezentującego jeden automat, na tym samym komputerze. Sposób doboru procesów może być różny, np. wybranie procesów reprezentujących automaty o krótkich czasach wykonania procedur tranzycji lub też wybranie procesów, które komunikują się między sobą wyjątkowo intensywnie itp. O wyborze ostatecznej, najkorzystniejszej alokacji, zależnej wybitnie od konkretnego zadania symulacyjnego, muszą zadecydować próbne eksperymenty symulacyjne. Zauważmy, że przy dowolnych możliwościach alokacji procesów, zaalokowanie wszystkich procesów na jednym komputerze jest podobne do symulacji sekwencyjnej. Różnica polega jednak na odmiennych regułach symulacji, a rozstrzygnięcie kwestii która z reguł jest bardziej korzystna z punktu widzenia czasu trwania symulacji wymaga praktycznej weryfikacji.

Innym istotnym zagadnieniem jest możliwość dekompozycji procesu, reprezentującego automat na dwa lub więcej podprocesy alokowane na różnych komputerach. Zagadnienie takie jest istotne np. wtedy, gdy dany proces stanowi „wąskie gardło"

symucji (ma najdłuższe ze wszystkich procesów występujących w danej symulacji czasy wykonywania tranzycji, przez co spowalnia działanie innych procesów) lub gdy dany proces wymaga zbyt dużej wielkości dostępnej pamięci. W obu przypadkach przyczyną takiego stanu rzeczy mogą być same procedury tranzycji lub/i niezbędne analizy statystyczne (np. w sytuacji, gdy dany automat ma dużo punktów interakcji, każdy z indywidualną kolejką wejściową). Pierwszy przypadek wymaga rozwiązania nowego problemu badawczego w odniesieniu do automatów - dekompozycji tranzycji. W drugim przypadku właściwym rozwiązaniem jest wykorzystanie równoległości na poziomie procedur.

5.4. Wybór platformy programowej symulacji

Każdy z systemów operacyjnych dostępnych na komputery osobiste (DOS, WINDOWS'95, OS/2, UNIX itp.) może być użyty w komputerach tworzących środowisko rozproszone symulacji. W przypadku alokacji na jednym komputerze kilku procesów, korzystniejsze jest użycie systemów wielozadaniowych. Interesującym zagadnieniem jest ocena narzutu czasowego degradującego efektywność symulacji, wnoszonego nie tylko przez transmisję informacji między komputerami, ale także przez same systemy operacyjne.

Szczegóły realizacji komunikacji międzyprocesowej powinny być zrealizowane według ogólnie przyjętych (de facto) standardów, np. MPI (ang. Message Passing Interface) lub PVM (Parallel Virtual Machine) [14]. Standard PVM jest jednak dostępny tylko pod UNIX-em.

Innym ważnym problemem jest wybór docelowego języka programowania. Ze względu na przyjętą zasadę komunikacji międzyprocesowej, stosowanie języków programowania równoległego (np. CC++, Fortran M itp.) nie jest konieczne, wystarczy użycie języków programowania sekwencyjnego. Zgodnie z wynikami pracy [5] wygodne i efektywne jest użycie techniki programowania obiektowego. W cytowanej pracy użyto rozszerzenia obiektowego języka Modula 2 [15]. Innymi językami, które powinny być brane pod uwagę to: C++, Java lub Turbo Pascal.

6. Kierunki dalszych prac

Zasadniczym celem rozwiązania problemów opisanych w p.5, wynikającym z zadania symulacji równoległej sieci rozszerzonych automatów asynchronicznych w środowisku rozproszonym, jest stworzenie wystarczających podstaw teoretycznych i praktycznych do budowy symulatora równoległego takich sieci.

Sieci automatów mogą być z powodzeniem stosowane do specyfikacji różnych systemów rozproszonych (obecnie prowadzone przez autora prace dotyczą specyfikacji modeli produkcyjnych i modeli ekonomicznych).

Szeroka klasa realnie występujących problemów, jaka może być modelowana przez sieci automatów oraz łatwo dostępna platforma, stanowiąca środowisko rozproszone symulacji - to podstawowe zalety takiego symulatora.

W pierwszej kolejności należy opracować symulator w dostępnym środowisku sieciowym, przyjmując algorytm asynchronicznej symulacji z wybranym protokołem zachowawczym oraz naturalną zasadą dekompozycji i alokacji procesów (jeden automat jest reprezentowany przez jeden proces, każdy proces jest alokowany na oddzielnym komputerze).

Dalsze prace powinny doprowadzić do wykonania symulatora równoległego, dającego użytkownikowi możliwość dokonania wyboru algorytmu symulacji oraz sposobu dekompozycji modelu i alokacji procesów, gwarantującego najlepszą efektywność symulacji w odniesieniu do każdego symulowanego modelu.

Literatura:

[1] Tyszer J.: Symulacja cyfrowa, WNT, Warszawa, 1990.
[2] New users' introduction to QNAP2 and QNAP2 reference manual, Simulog, 1984.
[3] Czachórski T. i inni: Modelowanie i ocena pracy systemów komputerowych za pomocą AMOK-u, Politechnika Śląska w Gliwicach, 1991.
[4] Dembinski P.: Using QNAP2 to evaluate performance of systems specified in Estelle. Rapport interne de recherche 91.06.07, TELECOM, 1991.
[5] Kacprzak M.: Metoda oceny charakterystyk czasowych projektów adapterów komunikacyjnych do sieci komputerowych. Rozprawa doktorska, Politechnika Śląska w Gliwicach, 1995.
[6] ISO 9074: Estelle: A formal description technique based on an extended transition model, 1989.
[7] ISO 7498: Information processing systems - Open systems interconnection - Basic reference model, 1984.
[8] ISO 7185: Programming language Pascal.
[9] Dembinski P., Budkowski S.: Simulating Estelle specification with time parameters. W książce Protocol specification, testing and verification VII, North-Holland, 1987.
[10] Ed.Lavenberg S.S.: Computer performance modeling handbook, Academic Press, 1983.
[11] Fujimoto R.M.: Parallel discrete event simulation, Communications of the ACM, 33(10), 1990.
[12] Ferscha A.: Parallel and distributed simulation of discrete event systems. W książce Ed. Zomaya A.Y.: Parallel and distributed computing handbook, McGraw-Hill, 1996.
[13] Słomiński L., Kaliszewski I.: Problemy równoległej optymalizacji dyskretnej, Polska Akademia Nauk, Instytut Badań Systemowych, Warszawa, 1994.
[14] Foster.I.: Designing and building parallel programs, Addison-Wesley, 1995.
[15] TopSpeed Modula 2. Language and library reference, Jensen & Partners International, 1991.

MAREK KOTOWSKI
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

SEZAM - System Ewidencji Zdarzeń
The SEZAM - events recording system

Streszczenie

W tekście przedstawiono opis podstawowej wersji programu SEZAM, zaprojektowanego i wdrożonego w Instytucie Maszyn Matematycznych. Opisane zostały elementarne pojęcia SEZAM-u, zwłaszcza obiekty i zdarzenia, oraz środki, jakie SEZAM dostarcza programiście aplikacyjnemu. Omówiono też potencjalne przyszłe kierunki rozwoju SEZAM-u.

Abstract

This article desribes the basic version of SEZAM - a program designed and implemented in the Instytut Maszyn Matematycznych. It deals with underlying concepts of SEZAM, especially objects, events and tools available for an application programmer. It also discusses the potential directions for SEZAM's growth.

Wstęp

SEZAM jest specjalizowanym programem, o charakterze zbliżonym do systemów bazy danych, przeznaczonym do tworzenia ukierunkowanych aplikacji. Pojawił się jako swoisty produkt uboczny pewnej grupy prac prowadzonych w Instytucie Maszyn Matematycznych, merytorycznie różnych, ale pod pewnym względem podobnych. Były to m.in. prace nad systemami ewidencji czasu pracy i kontroli dostępu, a także nad mikrokomputerowymi systemami pomiarowymi. Wszystkie one, jakkolwiek tematycznie różne, posługiwały się implicite podobnym modelem rzeczywistości: istniały podmioty, które czasowo wchodziły między sobą w relacje, ulegając w nich pewnym zmianom (zarówno same te relacje, jak i zmiany, jakie powodowały, były rejestrowane i raportowane). Pojawił się pomysł zaprojektowania narzędzia umożliwiającego programistom względnie łatwe tworzenie tego typu aplikacji, odpowiednio uniwersalnego, a jednocześnie efektywnego.

W tekście opisana jest wersja podstawowa programu SEZAM. Dla uproszczenia opisu pominięto niektóre, zaimplementowane już, bardziej złożone opcje programu. Prezentacja programu jest opisowa (o ewentualnym formalizmie opisu koncepcji, zwłaszcza definicji zdarzeń, mowa jest w końcowej części tekstu).

Przedstawiono główne struktury i prototypy funkcji SEZAM-u zakodowane w języku C (w tym języku została zaimplementowana podstawowa wersja SEZAM-u). W opisach dokumentów obiektu i zdarzeń, a także funkcji, użyto nazw angielskich. Jest to z jednej strony zgodne z nazwami istniejącymi w modułach SEZAM-u, z drugiej związane ze standardem kodowania obowiązującym w zespole w IMM, w którym SEZAM powstał.

1. Obiekty i zdarzenia

Każdy program może być traktowany jako model rzeczywistości, nawet jeśli tylko oblicza silnię (model iteracyjny i rekursywny obliczeń oznaczają różny sposób ich „widzenia", a dyskusje o wyższości jednego modelu nad drugim są tradycyjnym już problemem rozważanym na kursach programowania i w grupach dyskusyjnych Internetu). Również SEZAM oparty jest na pewnym określonym modelu rzeczywistości. Postrzega on świat przez obiekty i zdarzenia, w których te obiekty mogą brać udział i w których ulegają - czy też mogą ulegać - zmianom. Mówienie o modelu nie jest przy tym czystą retoryką - taki sposób traktowania programu pozwala określać nieformalnie potencjalny zakres aplikacji, do których oprogramowania można używać SEZAM-u.

Z punktu widzenia SEZAM-u obiekt jest dokumentem o pewnej strukturze. Wszystkie obiekty opisane przez dokument o tej samej strukturze tworzą zbiór obiektów. Również zdarzenie jest dokumentem o odpowiedniej strukturze. Przy czym sama nazwa „zdarzenie" oddaje dobrze sens tego, co denotuje: jest to zaistnienie w czasie pewnej relacji, np. między obiektami (zdarzenie może też być niezwiązane z żadnymi obiektami). Niech zobrazuje to przykład z trzema zbiorami obiektów: O1, O2, O3. Oto przykładowe zdarzenia, zapisane umownie, jako funkcje z argumentami:

Event1(),
Event2(O1),
Event3(O1),
Event4(O1,O1),
Event5(O2),
Event6(O1,O2),
Event7(O1,O2,O3).

Zdarzenie Event1 jest niezależne od jakichkolwiek obiektów. W dwóch następnych zdarzeniach bierze udział jeden obiekt ze zbioru O1, a w czwartym dwa obiekty z tego zbioru (ponieważ są to zdarzenia różnego typu, mogą też w różny sposób zmieniać obiekty biorące w nich udział). W zdarzeniu Event5 bierze udział obiekt ze zbioru O2, w Event6 - obiekty ze zbiorów O1 i O2, a w Event5 - obiekty ze zbiorów O1, O2, i O3.

Opis zdarzeń i to co znaczy „brać udział w zdarzeniu" i w jaki sposób zdarzenie może modyfikować obiekty, opisane jest niżej.

SEZAM umożliwia:

- definiowanie obiektów,
- dodawanie, usuwanie, modyfikowanie obiektów,
- definiowanie zdarzeń,
- realizowanie, przechowywanie i wyszukiwanie zdarzeń,
- odtwarzanie stanu obiektów z określonej chwili,
- obliczanie czasu trwania określonych stanów obiektów.

Wszystkie informacje definiujące obiekty i zdarzenia przekazywane są SEZAM-owi za pomocą odpowiednich struktur, które są argumentami odpowiednich funkcji SEZAM-u.

W wersji podstawowej SEZAM opracowano jako bibliotekę modułów. Powoduje to, że SEZAM jest integralną częścią aplikacji. Teoretycznie możliwe jest również, by SEZAM pracował w systemie jako niezależny proces, z którym programy aplikacyjne komunikowałyby się za pomocą odpowiednich środków, np. poprzez kolejki komunikatów lub potoki (patrz p. 8.5).

2. Opis obiektu

Jak już zaznaczono, obiekt jest dokumentem o określonej strukturze; wszystkie obiekty należące do jednego zbioru obiektów opisane są przez dokument o identycznej strukturze. Definiowanie zbioru obiektów jest faktycznie równoważne definiowaniu struktury dokumentu, który będzie opisywał obiekty należące do tego zbioru. Każdy zbiór obiektów ma określony numer (liczba całkowita), który go jednoznacznie identyfikuje. W strukturze opisującej obiekt zawsze muszą wystąpić dwa pola:

Status - opisuje stan obiektu (1 oznacza, że obiekt żyje, 0 - że jest usunięty),
LifeTime - opisuje czas życia obiektu.

Oprócz tych dwóch pól w definicji struktury dokumentu obiektu mogą występować - i zazwyczaj występują - pola, które mają różny typ i klasę pamięci, co będzie wyjaśnione niżej (wprawdzie Status i LifeTime również zostały nazwane umownie polami, niemniej ich struktura jest identyczna we wszystkich obiektach). Polom tym nadawane są wartości. Istotną cechą SEZAM-u jest to, że wszystkie wartości pól, przechowywane w dokumencie, bieżące i wcześniejsze (patrz niżej) mają jednocześnie zapisane czasy nadania tych wartości lub zakresy czasowe ich obowiązywania. Nie ma wartości pola nadanej poza czasem. Wszystkie wartości czasu, jakkolwiek różnie pakowane i zapisane w dokumentach obiektów czy zdarzeń, mają z punktu widzenia aplikacji taką samą postać: rok/miesiąc/dzień/godzina/minuta/sekunda.

Obiekty ułożone są w zbiorze (poindeksowane) według wartości jednego z pól. W wersji podstawowej SEZAM-u pole to jest określane w trakcie tworzenia aplikacji). Porządek ułożenia obiektów w zbiorze ma znaczenie dla funkcji pobierania obiektów (np. GetFirstObj, GetNextObj, patrz p. 2.2). W każdym zbiorze obiektów jeden obiekt jest obiektem bieżącym - na nim realizowane są np. zmiany pól (patrz p. 2.2).

2.1. Opis pola

Każde pole ma 7 atrybutów. Cztery pierwsze z nich są dość typowe dla pakietów przetwarzania dokumentów zawierających pola. Są to:

- numer (implicite 0-n),
- Type - Typ (int lub char),
- Length - Maksymalna długość pola (odnosi się do pola typu znakowego i wynosi 40 znaków),
- Range - odnosi się tylko do pola będącego liczbą całkowitą i zawiera dwie wartości określające jego zakres: najmniejszą i największą.

Podano tu dwa typy pola: int i char; w wersji podstawowej SEZAM-u są też inne pola (np. long, float), ale tu dla uproszczenia opisu pominięto je.

Trzy następne atrybuty pola w obiekcie są ściśle związane ze specyfiką SEZAM-u. Są to klasy pamięci oraz atrybuty Unique i Cleaning.

2.1.1. Klasy pamięci

Klasa pamięci pola określa sposób, w jaki przechowywane są w polu jego wartości. W wersji podstawowej SEZAM-u istnieją dwie klasy pamięci: AUTO i MEMO, przy czym ta ostatnia dzieli się na dodatkowe podklasy: MEMO_IND i MEMO_VAR (dalej będą one zwane po prostu klasami MEMO_IND i MEMO_VAR).

Jeśli pole ma klasę pamięci AUTO, to przechowywana w nim jest tylko ostatnia nadana mu wartość oraz czas jej nadania (w wersji podstawowej SEZAM-u pole to zmieniane jest tylko przez zdarzenia). Jeśli pole ma klasę pamięci MEMO, przechowuje wszystkie nadane mu wartości wraz z zakresem czasowym ich obowiązywania. Jeśli klasą pola jest MEMO_IND, to w danym momencie tylko jedna wartość pola jest bieżąca. Może ona się zmieniać w czasie (w polu przechowywane są wszystkie jego wartości, jakie przyjmowało), ale zawsze tylko jedna wartość jest bieżąca. Oznacza to także, że czasy trwania poszczególnych wartości pola klasy MEMO_IND nie mogą na siebie zachodzić (SEZAM kontroluje zachowanie tej reguły przy zmianie wartości pola klasy MEMO_IND i w miarę potrzeby łączy dwa zachodzące na siebie zakresy trwania tej samej wartości pola).

Pole klasy MEMO_VAR tym różni się od pola klasy MEMO_IND, że może mieć jednocześnie kilka wartości bieżących. Prostą konsekwencją tej zasady jest, że czasy trwania różnych wartości w polu klasy MEMO_VAR mogą na siebie zachodzić.

Gdyby wszystkie pola w dokumencie miały klasę MEMO, o wszystkich wartościach pól, przeszłych i obecnych, można by wnioskować z samego dokumentu obiektu. Ponieważ pole klasy AUTO zawiera tylko ostatnią wartość danego pola, zatem dla odtworzenia jego poprzednich wartości należy zrealizować ponownie odpowiednie zdarzenia.

Sam podział pól na klasy AUTO i MEMO jest pewnym założeniem modelowym. Odpowiada on - ogólnie rzecz biorąc - podziałowi na stany bardziej trwałe i szybkozmienne pól w obiekcie. W pewnym sensie jest to równoważne sposobom pamiętania stosowanym przez człowieka: część rzeczy pamiętamy, a część musimy sobie przypominać, odtwarzając w pamięci ciąg kolejnych zdarzeń, który doprowadził do danego stanu.

2.1.2. Atrybut Unique

Atrybut Unique (przybiera on dwie wartości: TRUE i FALSE) określa, czy wartość danego pola klasy MEMO ma być w danym momencie unikalna w zbiorze (tj. czy ma dany obiekt jednoznacznie identyfikować). Jeśli pole ma atrybut Unique równy YES - lub krócej i umownie: jeśli ma atrybut Unique - SEZAM sprawdza ten warunek przy uaktualnianiu pola. Jeśli próbuje się zmienić w obiekcie pole z atrybutem Unique na wartość, która w danym czasie występuje w tym samym polu w innym obiekcie (patrz p. 2.2.), modyfikacja nie jest realizowana i SEZAM sygnalizuje błąd.

Ogólny sens atrybutu Unique jest taki, że pozwala on na jednoznaczną identyfikację obiektu biorącego udział w zdarzeniu, przez jego cechę właściwą, a nie przez np. numer przydzielony przez SEZAM.

2.1.3. Atrybut Cleaning

Wartość pola klasy AUTO zmieniana jest przez zdarzenia. SEZAM umożliwia także nadawanie cyklicznie pewnych wartości polom klasy AUTO, niezależnie od zdarzeń. Operacja ta zwana jest odświeżaniem pola klasy AUTO. Odświeżanie pola definiuje się za pomocą struktury AUTO_CLEANING. Ma ona dwie składowe definiujące okres (co jaki czas pole jest odświeżane) oraz wartość nadawaną odświeżanemu polu:

```
typedef struct AutoTag
{
  int Period;  /* CO JAKI CZAS ODŚWIEŻAĆ POLE? */
  union
  {
    int IntValue;          /* NUMERYCZNA WARTOŚĆ POLA */
    char CharValue[20];    /* TEKSTOWA WARTOŚĆ POLA */
  } Value;      /* WARTOŚĆ DO WSTAWIENIA */
} AUTO_CLEANING;
```

W wersji podstawowej SEZAM-u okresami odświeżania są: godzina, dzień, tydzień, miesiąc. O tym, jak realizowane jest odświeżanie w wersji podstawowej SEZAM-u, mowa jest w rozdziale 4.

2.1.4. Struktura opisująca obiekt

Opiszemy teraz strukturę obiektu. Przedtem zdefiniujemy dwie dodatkowe struktury. Jak już zostało wspomniane, SEZAM pozwala definiować zakresy wartości pól numerycznych:

```
typedef struct RangeTag
{
  int IntRange[2];
} RANGE;
```

A oto struktura opisująca jedno pole:

```
typedef struct FieldTag
{
  int Type;      /* INT_OBJ, LONG_OBJ, CHAR_OBJ */
  int Length;    /* TYLKO DLA PÓL TEKSTOWYCH */
  int Memo;      /* YES, NO */
  int Unique;    /* YES, NO */
  AUTO_CLEANING AutoCleaning;
  RANGE Range;
} FIELD_DESC;
```

I opis calego zbioru obiektów:

```
#define FIELD_MAX  10
typedef struct ObjTag
{
  STATUS Status;          /* STATUS PLIKU */
  LIFE_TIME FileTime;     /* CZAS ŻYCIA OBIEKTU */
  int ObjHandle;          /* NUMER ZBIORU OBIEKTÓW */
  int NumberOfFields;     /* LICZBA PÓL W OBIEKCIE */
  int SortedBy;           /* NUMER POLA, WZGLĘDEM KTÓREGO PLIK JEST
                             SORTOWANY */
  FIELD_DESC FieldDesc[FIELD_MAX]; /* TABLICA Z OPISAMI PÓL */
} OBJECT_FILE;
```

Struktury te zostały tu nieco uproszczone (w istocie zamiast tablicy pól w strukturze OBJECT_FILE jest wskaźnik na obszar alokowany, zawierający daną tablicę). Jeśli składowa NumberOfFields w strukturze OBJECT_FILE jest równa 0, oznacza to, że obiekt nie ma pól (ma tylko typ i czas życia).

Oprócz tych struktur podamy jeszcze definicję struktury pomocniczej OBJECT_MASK

```
typedef struct ObjMaskTag
{
  int FieldFiled;    /* 0 - POLE NIEWYPEŁNIONE
                        1 - POLE WYPEŁNIONE */
  union
  {
    int Number;      /* WARTOŚĆ NUMERYCZNA */
    char Text[40];   /* WARTOŚĆ TEKSTOWA */
  } Value;
  unsigned long TimeInterval; /* CZAS TRWANIA STANU */
} OBJECT_MASK;
```

Struktura ta (tu również nieco uproszczona) jest nader użyteczna, jest bowiem środkiem wymiany informacji o obiektach między aplikacją a SEZAM-em. Między innymi wykorzystywana jest do określenia maski obiektów oraz do zwracania wartości pól (TimeInterval jest składową dla liczenia czasu trwania stanu i zawiera ostatnio obliczony czas trwania stanu, patrz p. 6.1).

2.2. Funkcje obsługi obiektów

SEZAM ma szereg funkcji obsługi obiektów przydatnych dla programisty aplikacyjnego. Pozwalają one pobierać i zmieniać dokumenty obiektów, a także wprowadzać do SEZAM-u nowe dokumenty. Duża część tych funkcji działa na obiekcie bieżącym (w zbiorze obiektów zawsze jeden obiekt jest obiektem bieżącym). Bardzo ważną cechą tych funkcji jest to, że ich argumentem jest czas. Oto te funkcje:

DefObj - Definiuje obiekt, a tym samym i zbiór obiektów,
AddObj - Dodaje obiekt do zbioru obiektów,
ChangeObj - Zmienia obiekt bieżący,

DeleteObj - Usuwa obiekt bieżący,
GetObj - Pobiera obiekt bieżący,
GetNextObj - Pobiera - i czyni bieżącym - następny obiekt w zbiorze,
GetPrevObj - Pobiera - i czyni bieżącym - poprzedni obiekt w zbiorze,
GetFirstObj - Pobiera - i czyni bieżącym - pierwszy obiekt w zbiorze,
GetLastObj - Pobiera - i czyni bieżącym - ostatni obiekt w zbiorze,
SetCurrObj - Czyni bieżącym obiekt o zadanych wartościach pól.

A oto przykładowy prototyp funkcji SetCurrObj

```
SetCurrObj(int ObjHandle, OBJ_MASK ObjMask[],
           DATE_TIME *DateTimePtr);
```

ObjHandle - Numer zbioru obiektów,
ObjMask - Tablica wartości poszczególnych pól,
DateTimePtr - Wskaźnik na strukturę zawierającą czas.

Jeśli wartością identyfikującą obiekt (ObjMask) jest wartość pola z atrybutem Unique, obiektem bieżącym zostanie obiekt, który ma (lub miał) taką właśnie wartość tego pola w czasie podanym w wywołaniu funkcji. Jeśli żadne z pól, których wartości podane są w ObjMask, nie ma atrybutu Unique, bieżącym dokumentem staje się pierwszy obiekt w zbiorze, którego wartości pól są zgodne z wartościami podanymi w ObjMask.

2.3. Obiekt w czasie

Pobieranie dokumentu obiektu jest w istocie operacją złożoną. Podając czas, z jakiego mają być pobrane wartości pól dokumentu obiektu, faktycznie wymusza się na SEZAM-ie wykonanie takich operacji, by wszystkie pola klasy AUTO w dokumentach obiektów miały takie wartości, jakie miały w danym czasie. Gdyby w dokumentach obiektów były tylko pola klasy MEMO, pobranie wartości każdego pola byłoby operacją realizowaną tylko w obrębie dokumentu obiektu (wszystkie wartości pól klasy MEMO znajdują się w dokumencie obiektu). Jeśli jednak dokument zawiera przynajmniej jedno pole klasy AUTO, do określenia wartości tego pola w danym czasie musi być na ogół zrealizowana odpowiednia sekwencja zdarzeń. A oto jak działa SEZAM przy wywołaniu np. funkcji GetObj z podanym, jako argumentem, pewnym czasem z przeszłości. SEZAM zatem:

- odnajduje moment wcześniejszy od podanego czasu, taki że wartości wszystkich pól AUTO są określone (praktycznie jest to moment odpowiadający najdłuższemu okresowi odświeżania),
- pobiera kolejno wszystkie zdarzenia od tego momentu do chwili określonej w wywołaniu funkcji GetObj i symuluje ich realizację. W trakcie realizowania zdarzeń przeprowadzane jest także odświeżanie pól klasy AUTO w obiektach zbioru, dla którego takie odświeżanie zdefiniowano.

Czas podany w wywołaniu funkcji GetObj jest przez SEZAM przechowywany. Jeśli następne wywołania, np. funkcji GetNextObj, będą zawierały ten sam czas, SEZAM nie będzie obliczał od nowa stanów obiektów (stany te są już obliczone), ale pobierał po prostu wartości pól obiektów. Jeśli czas w następnym wywołaniu funkcji

będzie o godzinę późniejszy, SEZAM zrealizuje wydarzenia między jednym a drugim czasem i obliczy stany obiektów w nowym czasie. Jeśli czas podany w nowym wywołaniu funkcji będzie wcześniejszy od czasu podanego w poprzednim wywołaniu, SEZAM ponowi całą procedurę.

Usuwanie obiektu ze zbioru polega na zmianie zawartości pól Status i LifeTime. Sam usunięty obiekt pozostaje nadal w zbiorze. Dzięki temu przy określaniu stanów obiektów w czasie, gdy obiekt istniał (nie był jeszcze usunięty), będzie on brany pod uwagę.

3. Zdarzenie - definicja

Zdarzenie jest opisane przez dokument o określonej strukturze. Sensem zdarzenia jest: a) sam fakt, że zaistniało, b) ewentualne zmiany, jakich zdarzenie dokonuje w obiektach biorących w nim udział (stosownie do definicji zdarzenia).

Zdarzenie jest dokumentem, który składa się z typu, czasu zaistnienia, składowych oraz akcji. Typ zdarzenia jest liczbą całkowitą (od 0 do 255) i razem z czasem zdarzenia występuje w każdym zdarzeniu. SEZAM, znając definicję zdarzenia danego typu, „wie", jak potraktować jego zawartość, tj. składowe, i jakie operacje (czyli akcje) zrealizować (jeśli SEZAM nie zna typu zdarzenia, w wersji podstawowej ignoruje je).

3.1. Składowe zdarzenia

Oprócz typu i czasu zdarzenia definicja zdarzenia może zawierać składowe zdarzenia, a także akcje. Składowe zdarzenia mogą być dwojakiego rodzaju: niezależne i obiektowe. W pierwszym przypadku składowa jest wartością niezależną niezwiązaną - formalnie rzecz biorąc - z jakimkolwiek obiektem (typ tej składowej w wersji podstawowej SEZAM-u może być int lub char). Oto definicja składowej niezależnej:

```
typedef union IndMemTag
{
  int Value;
  int Text[40];
} IND_MEMBER;
```

Drugim rodzajem składowej jest składowa zwana umownie obiektową. W definicji takiej składowej podany jest numer zbioru obiektów oraz numer pola w obiekcie. W wersji podstawowej SEZAM-u jest to zawsze numer pola z atrybutem Unique. Zadaniem tej składowej jest identyfikowanie obiektu występującego w zdarzeniu. Ponieważ wartość pola z atrybutem Unique może zmieniać się z czasem, ten sam obiekt może brać udział w zdarzeniach w różnym czasie i być identyfikowany przez różne wartości tego samego pola.

Zgodnie z tym co powiedziano w rozdziale 1, w jednym zdarzeniu może brać udział więcej niż jeden obiekt z jednego zbioru, a także obiekty z różnych zbiorów (każdemu obiektowi odpowiada jedna składowa). A oto definicja składowej obiektowej:

```
typedef struct ObjMemTag
{
  int ObjHandle;  /* NUMER ZBIORU OBIEKTÓW */
```

```
  int FieldNo;      /* NUMER POLA, KTÓREGO WARTOŚĆ WYSTĘPUJE W
                       ZDARZENIU */
} OBJ_MEMBER;
```

A oto definicja całej składowej zdarzenia:

```
typedef union MemTag
{
  IND_MEMBER IndMember;
  OBJ_MEMBER ObjMember;
} MEMBER;
```

3.2. Akcje zdarzenia

Zdarzenia mogą powodować zmiany w obiektach biorących udział w zdarzeniu. W podstawowej wersji SEZAM-u zakres tych zmian jest bardzo ograniczony, obejmując nadawanie polu wartości oraz dodanie/odjęcie stałej od wartości pola. A oto struktura, za pomocą której definiuje się akcję:

```
typedef struct
{
  int MembNo;       /* NUMER SKŁADOWEJ OBIEKTOWEJ IDENTYFIKUJĄCEJ
                       OBIEKT */
  int FieldNo;      /* NUMER MODYFIKOWANEGO POLA W OBIEKCIE */
  int Operation;    /* OPERATOR AKCJI: = + - */
  union
  {
    int IndMemNo;     /* NUMER SKŁADOWEJ NIEZALEŻNEJ */
    int  Value;       /* WARTOŚĆ LICZBOWA */
    char Text[20];    /* TEKST - TYLKO DLA OPERATORA = */
  } NewValue;
} ACTION;
```

Składowa IndMemNo została wprowadzona, aby wykorzystać w akcji wartości składowej niezależnej. Faktycznie jest to rodzaj protezy, wprowadzonej do wersji podstawowej i stosowanej w operacjach na polach obiektu z wartościami składowymi niezależnymi (w następnych wersjach SEZAM-u pojęcie wartości biorącej udział w operacji na zdarzeniu zostanie odpowiednio uogólnione).

Skoro mamy zdefiniowane składowe i akcje, możemy zdefiniować całe zdarzenie:

```
#define MEMB_NUMBER 4
#define ACT_NUMBER 4

typedef struct EventTag
{
  int Type;              /* TYP ZDARZENIA */
  DATE_TIME DateTime;  /* CZAS ZDARZENIA */
  MEMBER Members[MEMB_NUMBER];    /* SKŁADOWE ZDARZENIA */
  ACTION Actions[ACT_NUMBER];     /* AKCJE ZDARZENIA */
} EVENT_DEF;
```

W wersji podstawowej SEZAM-u zmiany w polach obiektu odbywają się bezwarunkowo i tylko na obiektach biorących udział w zdarzeniu. Jeśli w zdarzeniu jest tylko jedna składowa obiektowa, mogą być modyfikowane/nadawane wartości tylko polom obiektu określonego przez tę składową.

SEZAM przyjmuje definicje zdarzeń oraz same zdarzenia i realizuje zmiany obiektów, jakie wywołują. Przechowuje także zdarzenia i umożliwia ich wyszukiwanie. Porządek czasowy wprowadzania zdarzeń nie ma znaczenia - można w danym momencie wprowadzić zdarzenie sprzed miesiąca lub roku. Jeśli zdarzenie jest wcześniejsze niż czas ostatniej modyfikacji pola klasy AUTO w obiekcie, zdarzenie nie modyfikuje tego pola. Jeśli czas zdarzenia jest późniejszy niż czas modyfikacji pola klasy AUTO, zdarzenie jest realizowane (przez modyfikacje pola klasy AUTO rozumie się także jego odświeżanie). Tak jest w wersji podstawowej SEZAM-u. Przy bardziej złożonych definicjach systemu sytuacja może nie być tak klarowna (patrz p. 8.5).

4. Zdarzenie zewnętrzne i wewnętrzne

Zdarzenia opisane wyżej są domyślnie generowane przez aplikację. Mogą pojawiać się w dowolnym czasie i kolejności. Zdarzenia takie nazywane są zewnętrznymi. Oprócz nich w SEZAM-ie można zdefiniować zdarzenia wewnętrzne, pojawiające się niezależnie od aplikacji. Innymi słowy: SEZAM, generując zdarzenia wewnętrzne, może „żyć swoim własnym życiem", niezależnie od świata zewnętrznego. Jest to cecha programu bardzo atrakcyjna, ale jednocześnie szczególnie trudna w implementacji.

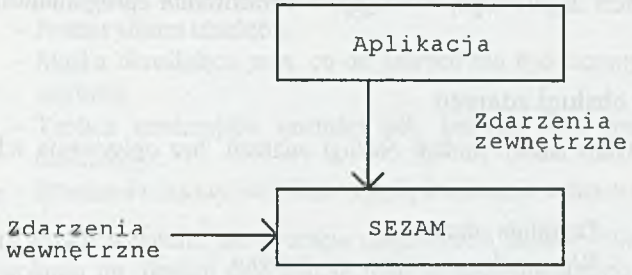
W wersji podstawowej SEZAM-u zdarzenia wewnętrzne są dość ograniczone. Definicja zdarzenia wewnętrznego ma nieco inną strukturę niż definicja zdarzenia zewnętrznego:

```
typedef struct InEventTag
{
  int Type;          /* TYP ZDARZENIA */
  int Period;        /* CO ILE CZASU ZDARZENIE SIĘ POWTARZA */
  IN_ACTION Action[4]; /* AKCJE ZDARZENIA */
} INNER_EVENT;

typedef struct
{
  int FileHandle;   /* NUMER ZBIORU OBIEKTÓW */
  int FieldNo;      /* NUMER POLA */
  int Operation;    /* OPERATOR AKCJI: = + - */
  union
  {
    int  Number;    /* WARTOŚĆ BIORĄCA UDZIAŁ W OPERACJI, NP.
                       PRZYPISANIA  */
    char Text[20]; /* TYLKO DLA OPERACJI PRZYPISANIA */
  } NewValue;
} IN_ACTION;
```

Dodatkowa istotna różnica między zdarzeniem zewnętrznym i wewnętrznym w podstawowej wersji SEZAM-u polega na tym, że zdarzenia wewnętrzne dotyczą wszystkich obiektów danego zbioru (nie można realizować zdarzenia wewnętrznego

tylko na jednym wybranym obiekcie). Zdarzenia te są również brane pod uwagę przy obliczaniu stanów obiektów (SEZAM realizując zdarzenia niezbędne dla określenia stanu obiektów w danym czasie, realizuje także w odpowiednich momentach zdarzenia wewnętrzne, o ile zostały one zdefiniowane).



Rys. Zdarzenia zewnętrzne i wewnętrzne

Trzeba powiedzieć, że ze zdarzeniami wewnętrznymi jest kłopot. Sposób realizacji zdarzenia wewnętrznego jest - czy też może być - silnie zależny od środowiska operacyjnego. W wersji podstawowej SEZAM-u zdarzenia wewnętrzne są symulowane. Kiedy wywołana zostaje jakakolwiek funkcja SEZAM-u, sprawdza on, czy w międzyczasie, tj. między poprzednim a obecnym wywołaniem, nie powinno pojawić się zdarzenie wewnętrzne. Jeśli tak, generuje je i stosownie zmienia wartości pól w obiektach. Notabene w ten sam sposób realizowane jest w wersji podstawowej SEZAM-u również odświeżanie wartości pól.

Taki sposób realizacji zdarzeń wewnętrznych i odświeżania pól klasy AUTO ma tę zaletę, że jest przenośny (nie wykorzystuje żadnych specjalnych opcji środowiska, w którym pracuje). Ale ma wady: jest nieefektywny - przy każdym odwołaniu do SEZAM-u musi być realizowane w takiej czy innej formie sprawdzanie, czy nie należy odświeżyć wartości pól lub wygenerować zdarzenia wewnętrznego. Wadą poważniejszą tego rozwiązania (wersji podstawowej SEZAM-u to akurat nie dotyczy) jest ograniczenie możliwości samodzielnego zachowania się SEZAM-u. W planowanych rozszerzeniach programu przewidywana jest możliwość generowania przez SEZAM komunikatów lub nowych zdarzeń (patrz p. 8.5). Jeśli możliwości takie miałyby być wykorzystane przy realizacji zdarzeń wewnętrznych (np. SEZAM miałby w pewnych sytuacjach co godzinę generować komunikat), w wersji z symulowaniem zdarzeń wewnętrznych byłoby to niemożliwe, a przynajmniej utrudnione - SEZAM musiałby otrzymywać w odpowiednich momentach sterowanie. Tym samym czuwanie nad właściwą realizacją zdarzeń wewnętrznych spoczywałoby na aplikacji, co byłoby sztuczne i sprzeczne z ogólną zasadą działania SEZAM-u.

Techniczna realizacja zdarzeń wewnętrznych wymaga wprowadzenia tego, co żargonowo nazywa się „wejściem zegarowym". W systemie operacyjnym MS-DOS oznacza to przejęcie przerwania zegarowego. Realizacja tego może nastręczać pewne trudności. Otóż w funkcjach SEZAM-u przejmujących przerwanie zegarowe niezbędne może być zrealizowanie pewnych odwołań do systemu. Ponieważ nie wiadomo, jakie operacje były wykonywane w systemie w momencie pojawienia się przerwania

zegarowego, konieczne jest sprawdzenie, czy np. nie była wykonywana jakaś funkcja systemowa. Można tu wykorzystać tzw. stan jałowy systemu, w którym - mimo że DOS jest aktywny - można wywoływać funkcje systemowe.

Takie rozwiązanie w systemie MS-DOS będzie działać, ale oczywiście jest wysoce zależne od środowiska. W innym środowisku może funkcjonować inna zasada tworzenia wejścia zegarowego, wymagająca opracowania oprogramowania tej funkcji od nowa.

4.1. Funkcje obsługi zdarzeń

Podamy tylko nazwy funkcji obsługi zdarzeń, bez opisywania ich argumentów i struktur:

DefEvent - Definiuje zdarzenie,
StoreEvent - Wprowadza zdarzenie zewnętrzne,
Get/SetEvent - Pobiera/wybiera zdarzenie przechowane,
GetNextEvent - Pobiera następne przechowane zdarzenie,
GetEvenInfo - Pobiera informacje statystyczne o zdarzeniach.

Pierwsza z nich zawiera pełny opis zdarzenia, pozostałe - tylko poszczególne składowe, zapisane w odpowiedniej strukturze.

5. Definiowanie systemu aplikacyjnego

Definiowanie całego systemu wykorzystującego SEZAM polega na zdefiniowaniu obiektów oraz zdarzeń. W większości tworzonych aplikacji definicje zdarzeń i obiektów będą określane z góry, czyli wprowadzane do odpowiedniego zbioru definicyjnego, który SEZAM będzie wczytywał na początku sesji pracy (do aplikacji nie będą również włączane moduły zawierające funkcje przyjmujące definicje obiektów i zdarzeń). Formalnie rzecz biorąc, można utworzyć system, który będzie dynamicznie przyjmował nowe definicje zdarzeń i obiektów. Aplikacja taka musiałaby jednak być na tyle elastyczna, by móc takie zdarzenia i obiekty obsługiwać, np. formatować raporty dokumentów o zmiennej strukturze.

6. Raporty SEZAM-u

SEZAM pozwala uzyskiwać informacje o obiektach, za pomocą funkcji wymienionych w poprzednich rozdziałach (np. GetFirstObj, GetNextObj). Wprowadzają one implicte obiekty w stan odpowiadający danemu czasowi. Za pomocą tych funkcji można zbudować raport o stanie obiektów z dowolnego czasu.

Można również uzyskiwać informacje o zdarzeniach za pomocą odpowiednich funkcji (np. Get/SetEvent, GetNextEvent). Jest wszakże jeden raport w SEZAM-ie, wydzielony i szczególnie ważny, który tworzy się za pomocą oddzielnej funkcji i w którym ujawniają się praktyczne zalety SEZAM-u. Jest to raport o czasie trwania stanów obiektów.

6.1. Raport o czasie trwania stanów

Informacje o czasie trwania określonych wartości pól pozwala uzyskać funkcja CountStateIntervals. Oto jej prototyp:

```
void CountStateIntervals(int ObjHandle,
                         OBJ_MASK ObjMask[],
                         OBJ_VALUE ObjValue[],
                         TIME_RANGE *TimeRangePtr);
```

A oto opis argumentów:

Handle - Numer zbioru obiektów,
ObjMask - Maska określająca pola, co do których ma być liczony czas trwania wartości,
ObjValue - Tablica zawierająca wartości pól, których czas trwania ma być obliczony,
TimeRange - Przedział czasowy, w jakim mają być mierzone czasy trwania stanów.

Funkcja ta działa podobnie jak operacja znajdowania stanów obiektów, realizowana przy wywołaniu np. funkcji GetFirstObj, tyle że liczy dodatkowo czasy trwania wybranych stanów. Obliczone czasy są umieszczane w polach dynamicznych alokowanych dla obiektów. I programista musi je pobrać za pomocą np. funkcji GetFirstObj i GetNextObj (przy pobieraniu obiektu czasy te są umieszczane w składowych TimeInterval struktury OBJECT_MASK). W przeciwnym razie następne wywołanie funkcji CountStateIntervals zamaże poprzednie obliczone wartości czasów trwania stanów.

7. Przykłady

SEZAM może być użyty do tworzenia aplikacji, które dadzą się sprowadzić - chociażby w pewnym rozsądnym przybliżeniu - do wspomnianego modelu obiektów i zdarzeń, w których te obiekty ulegają zmianie (pomijamy tu sprawę ograniczeń istniejących w podstawowej wersji SEZAM-u). Wbrew pozorom jest to bardzo liczna i różnorodna grupa aplikacji. Omówimy tylko dwa przykłady. Są całkowicie różne i podane tu dla zilustrowania uniwersalności koncepcji.

7.1. System rejestracji czasu pracy

Oto krótki opis najprostszej wersji systemu rejestracji czasu pracy zaprojektowanej z wykorzystaniem SEZAM-u. Obiektem jest pracownik, który opisany jest przez dokument o 5 polach:

Pole 0 First Name - klasa MEMO_IND
Pole 1 Second Name - klasa MEMO_IND
Pole 2 Code (Unique) - klasa MEMO_IND
Pole 3 Status - klasa MEMO_IND
Pole 4 Presence - klasa AUTO

Cztery pierwsze pola dokumentu obiektu są typu znakowego i klasy MEMO_IND, ostatnie pole - typu int i klasy AUTO (ma ono okres odświeżania równy 24 godzinom). Pole Status oznacza stan pracownika (praca, urlop, zwolnienie lekarskie etc.). Zdefiniowane są dwa najprostsze zdarzenia. Pierwszym jest wejście, drugim wyjście z zakładu pracy. Każde z tych zdarzeń ma typ, czas oraz jedną składową obiektową - zawartość pola Code, będącego numerem zapisanym np. na karcie magnetycznej. Każde zdarzenie ma też jedną akcję, która zmienia zawartość pola Presence w

dokumencie obiektu na wartość PRESENT (wejście) lub ABSENT (wyjście). Raport o obecności pracowników realizuje się pobierając po prostu ich dokumenty z podaniem odpowiedniego czasu (pole Presence będzie zawierać informację o tym, czy pracownik był w danej chwili obecny w zakładzie pracy). Ogólny czas pracy, potrzebny np. do sporządzenia listy płac, obliczany jest przez wywołania funkcji CountStateIntervals (w nawiasach podane są umownie pola i wartości tych pól, których czasy trwania są liczone):

```
WorkTime= CountStateIntervals(Presence==PRESENT)+
          CountStateIntervals(Status==HOLLIDAY)+
          CountStateIntervals(Status==SICK);
```

Takie wyrażenie (zakodowane tu umownie i skrótowo) spowoduje obliczenie w danym przedziale czasowym (tu pominiętym) łączny czas pracy pracownika zsumowany z czasami jego pobytu na zwolnieniu lekarskim i na urlopie.

7.2. System pomiarowy

Drugim przykładowym systemem wykorzystującym SEZAM niech będzie system pomiarowy, np. pomiaru temperatury. Obiektami są tu urządzenia pomiarowe, a jedno z pól klasy AUTO w dokumencie każdego urządzenia - wartością ostatniego pomiaru. Zdarzeniem jest sam pomiar. Ma on dwie składowe: numer urządzenia oraz wartość pomiaru (składowa niezależna). Po wykonaniu pomiaru wartość składowej niezależnej jest wpisywana do pola klasy AUTO w dokumencie urządzenia pomiarowego. Raport o temperaturze z dowolnego dnia i godziny jest realizowany przez pobranie dokumentów obiektów z danego czasu, zaś raport o czasie np. trwania danej temperatury - przez wywołanie funkcji CountStateIntervals z odpowiednią wartością pola klasy AUTO.

8. Dalsze prace

Przedstawiono zasady działania podstawowej wersji SEZAM-u. Jest to wersja bardzo prosta, w opisie dodatkowo uproszczona. Może ona wprawdzie służyć do tworzenia prostych systemów, niemniej jej użyteczność jest ograniczona. Większe możliwości dadzą następne wersje SEZAM-u.

SEZAM może być rozszerzany w różnych kierunkach. Opiszemy teraz możliwości jego rozwoju. Są różne, zarówno co do charakteru, jak i użyteczności i oczywiście nakładu pracy, jakiego wymagają.

8.1. Zmiany ogólne

W wersji podstawowej SEZAM jest biblioteką modułów dołączoną do aplikacji, a wszelka komunikacja z nim realizowana jest poprzez wywołania funkcji z odpowiednimi argumentami. Sposób działania SEZAM-u można poszerzyć, np.:

- wprowadzić formalizm opisu SEZAM-u, a zwłaszcza obiektów i zdarzeń, co umożliwiłoby łatwe programowanie SEZAM-u;
- zmienić interfejs aplikacyjny na kolejkę komunikatów, co zwiększyłoby uniwersalność stosowania SEZAM-u;
- umożliwić obsługę przez SEZAM kilku procesów działających równolegle na niekoniecznie rozłącznych zbiorach obiektów.

8.2. Obiekty

Środki definiowania i przetwarzania samych obiektów można też uzupełnić i poszerzyć. Można zatem:

- wprowadzić nazwę pola (np. dla łatwiejszego definiowania zdarzeń);
- wprowadzić więcej typów pól i zwiększyć ich maksymalne długości;
- umożliwić dynamiczną zmianę pola, według którego obiekty w zbiorze są sortowane;
- wprowadzić pojęcie grupy obiektów w zbiorze i stworzyć składową grupową w zdarzeniach;
- umożliwić realizację odświeżania pól klasy AUTO za pomocą funkcji zdefiniowanej zewnętrznie (adres tej funkcji definiuje SEZAM-owi programista aplikacyjny).

8.3. Zdarzenia

Bez wątpienia największe możliwości kryją się w potencjalnym rozszerzeniu struktury i funkcji zdarzeń. W wersji podstawowej SEZAM-u akcje w zdarzeniu odnoszą się tylko do obiektów biorących udział w zdarzeniu i ograniczają się do bezwarunkowych zmian zawartości pól tych obiektów. A oto inne dodatkowe akcje, jakie można by wprowadzić do zdarzeń:

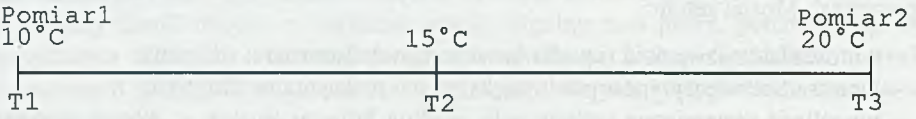
- usuwanie obiektu/obiektów biorących udział w zdarzeniu,
- dodanie nowego obiektu do zbioru,
- operacje wieloobiektowe (grupowe),
- generowanie komunikatów do aplikacji,
- operacje na obiektach nie biorących udziału w zdarzeniu,
- generowanie nowych zdarzeń.

Jest to lista możliwych rozszerzeń akcji. Można również wprowadzić inne zmiany, np. poszerzyć akcje o zdanie warunkowe if-else, czy wprowadzić opcję UNDO, która umożliwiałaby anulowanie skutków ostatniego zdarzenia. Można uzależniać zmianę pola od wartości wcześniejszych albo późniejszych innych pól tego samego obiektu (implementacja takiej opcji w odniesieniu tylko do pól klasy MEMO nie jest trudna - z polami klasy AUTO jest gorzej). Można też wprowadzić parametr określający rodzaj wejścia zegarowego dla odświeżania pól i generowania zdarzeń wewnętrznych (np. czy musi być synchroniczne, czy może być tylko symulowane).

8.4. Rozszerzenia raportów

W wersji podstawowej SEZAM-u raporty opisujące stany obiektów uwzględniają jedynie ostatnie wartości nadane polom klasy AUTO przed czasem, w którym raport jest sporządzany. W systemie pomiarowym temperatury (patrz rysunek niżej) wartość raportowana w czasie T2, znajdującym się między czasami dwóch sąsiednich pomiarów, np. realizowanych co godzinę, będzie równa 10°C, mimo że najwyraźniej temperatura wzrasta i w rzeczywistości w czasie T2 będzie wyższa niż w T1. Lepszym wyjściem byłoby ekstrapolowanie stanów obiektów (np. przez proste uśrednianie arytmetyczne wartości stanów sąsiadujących na osi czasu). W takiej sytuacji wartość pola

klasy AUTO w obiekcie (temperatura) w czasie T2 wynosiłaby np. 15°C (zakładając, że T2 byłoby równe T1 + (T3-T1) 0,5).



Bardziej elastycznym rozwiązaniem byłoby umożliwienie deklarowania funkcji obliczającej wartości stanów pośrednich w polach klasy AUTO (adres takiej funkcji byłby określony przez programistę aplikacyjnego).

Raport o czasie trwania stanów pól w obiektach można i należy poszerzyć o relacje. Sprawdzane zatem mogłoby być nie tylko to, ile czasu pole miało określoną wartość, ale także jak długo prawdziwa była relacja zachodząca dla tej wartości (np. wartość temperatury była większa lub mniejsza od wartości zadanej).

8.4.1. Symulacje alternatywnych stanów obiektów

W wersji podstawowej SEZAM, określając stany obiektów w zadanym czasie, uwzględnia wszystkie zdarzenia, jakie zostały zarejestrowane od odpowiedniego momentu, (jest on określony przez najdłuższy okres odświeżania) do danego czasu. W dowolnym zadanym czasie stan obiektów będzie zatem zawsze taki sam. Innymi słowy: w wersji podstawowej SEZAM zawsze opisuje stany obiektów tak, jak się miały w rzeczywistości. Można wprowadzić mechanizm, który pozwalałby wybierać selektywnie zdarzenia, jakie mogą być brane pod uwagę przy określaniu stanów obiektów. W najprostszym przypadku mogłyby to być tylko typy zdarzeń, jakie należy uwzględniać. Dodatkowo można wprowadzić inne rodzaje ograniczeń. Mogą one odnosić się tylko do czasu, np. brane pod uwagę będą tylko zdarzenia zachodzące we wtorki i w czwartki, między godziną 9 a 9.15, lub odwrotnie: określić zakres czasowy, z którego zdarzenia nie będą brane pod uwagę (w szczególności mogłoby to być jedno wybrane krytyczne zdarzenie, które zmieniło radykalnie stan obiektów). Można też uwzględniać tylko zdarzenia następujące po sobie w czasie nie krótszym lub nie dłuższym niż określony interwał czasowy. Można by pójść dalej: w definiowaniu zdarzeń, jakie byłyby brane pod uwagę przy określaniu stanu obiektów, uzależnić je od pewnych wartości wybranych pól. Określiwszy taki zbiór ograniczeń na zdarzenia, można generować różnorakie sekwencje stanów, prowadząc badania typu „co by było, gdyby".

8.5. O niebezpieczeństwach ingerowania w przeszłość

Rozszerzenie możliwości SEZAM-u, a zwłaszcza rodzajów akcji realizowanych w zdarzeniach, może prowadzić do powstania nowych problemów, z jednej strony dość oryginalnych, a z drugiej trudnych do rozwiązania. Co zrobić, jeśli zdarzenie o czasie wcześniejszym od bieżącego, wprowadzone do SEZAM-u, zmieni radykalnie stan bieżący (tj. spowoduje zmiany w obiektach w sposób wprawdzie zgodny z logiką zdefiniowanych zdarzeń, ale sprzeczny z intencjami, czy ze zdrowym rozsądkiem)? Jest to klasyczny problem z utworów fantastyczno naukowych. Oto typowa historia. W przyszłości (dalekiej?) nauczono się podróżować w czasie. Powstały firmy, które oferowały

podróże w przeszłość, by można było zobaczyć, jak żyli przodkowie. Podróże odbywały się w specjalnym pojeździe, który uniemożliwiał podróżnikom kontakt z obserwowaną rzeczywistością. Podstawowym założeniem było, że nie można w żaden sposób ingerować w przeszłość. Niestety, jeden z podróżników w trakcie wyprawy w świat wcześniejszy o 1000 lat wyłamał się z tej zasady - opuścił pojazd i zabił przypadkiem motyla. Kiedy wycieczka wróciła w teraźniejszość, okazało się, że zmiany spowodowane śmiercią motyla skumulowały się przez wieki w istny wicher historii. Po świecie, który podróżnicy w czasie opuścili, zostały tylko zgliszcza...

Podobne wydarzenie - mowa oczywiście o sensie zjawiska, a nie o jego wadze czy skali - może zdarzyć się w SEZAM-ie. W przypadku gdy dodane zdarzenie z przeszłości, np. grupowe, będzie powodować dodawanie lub usuwanie obiektów, przy odpowiedniej definicji zdarzeń może to pociągnąć za sobą łańcuch operacji, po których - w krańcowym przypadku - np. większość obiektów zniknie, pojawi się ich bardzo dużo, wszystkie wartości pól zostaną wyzerowane etc. Słowem - wprowadzenie zdarzenia z przeszłości może okazać się destrukcyjne dla całości, a tym samym niepożądane.

Oczywiście problemy tego typu będą pojawiały się w bardzo specyficznych sytuacjach. W opisanym wyżej systemie rejestracji czasu pracy problemów takich nie będzie - zdarzenia zmieniają jedynie zawartości pól Presence w obiektach, które w dodatku co 24 godziny są odświeżane. Kiedy jednak z jednej strony zrealizowane zostaną rozszerzenia SEZAM-u, z drugiej zaś utworzone zostaną aplikacje o dość rozbudowanych definicjach zdarzeń - problemy z niepożądanymi skutkami zmian wprowadzanych w zdarzeniach z przeszłości mogą się pojawić.

Można zastosować różne rozwiązania. Krańcowym jest albo w ogóle nie wprowadzać zdarzeń wcześniejszych niż chwila bieżąca (byłoby to równoznaczne z rezygnacją z bardzo użytecznych środków dostarczanych przez SEZAM), albo ignorować skutki wprowadzenia zdarzeń z przeszłości (tylko je rejestrować). Raporty o zdarzeniach uwzględniałyby takie zdarzenia, ale raporty o stanach obiektów - nie. Rozwiązanie przeciwstawne, z drugiego końca spektrum, to iść ze zmianami do końca, ze wszystkimi skutkami, nawet z owymi „zgliszczami"...

Obydwa rozwiązania są stosunkowo proste w implementacji, ale złe z różnych przyczyn (pierwsze ogranicza programistę, drugie - daje mu zbyt mało kontroli nad zachowaniem się SEZAM-u). Można wybrać rozwiązania pośrednie, o różnym charakterze, ograniczające potencjalne zmiany w przeszłości. I tak można pozwolić na wprowadzanie zdarzeń wcześniejszych od chwili obecnej, ale tylko o pewien określony odcinek czasu, np. tydzień. Zdarzenia wcześniejsze o 8 dni od chwili obecnej nie byłyby przyjmowane przez SEZAM. Innym rozwiązaniem jest ograniczyć rodzaje zdarzeń, jakie mogą być rejestrowane z czasem wcześniejszym od bieżącego (np. tylko zdarzenia nie powodujące dodawania/usuwania obiektów lub nie działające na grupach obiektów). Można też wprowadzić swoiste „ściany czasowe". Byłyby one zapisem wszystkich stanów obiektów w danym momencie (ściślej: wartości pól klasy AUTO). SEZAM przyjmowałby dowolne zdarzenia z przeszłości i realizowałby je, ale tylko między jedną a drugą „ścianą czasową". Jakiekolwiek zmiany by te zdarzenia powodowały w przeszłości, „ściana czasowa" by je anulowała. Jeśli „ściany czasowe" tworzone byłyby np. o północy, dodanie zdarzenia z poprzedniego dnia z godziny 14.15 mogłoby powodować różne zmiany w obiektach, ale o godzinie 0.0 stan wszystkich obiektów byłby taki, jak stany obiektów zapisane w „ścianie czasowej". Wariant ten ma tę

dodatkową zaletę, że „ściana czasowa" jest jednocześnie punktem, od którego można zacząć realizację zdarzeń przy określaniu stanów obiektów. Wadą rozwiązania jest, że może ono uniemożliwiać albo utrudniać przeprowadzanie badań z wybiórczym traktowaniem zdarzeń z okresów dłuższych niż czasy między utworzeniem kolejnych „ścian czasowych".

Rozwiązaniem najlepszym byłoby symulować zmiany, jakie może wprowadzić dane zdarzenie w SEZAM-ie i w zależności od wyników takiej symulacji wprowadzać zdarzenie do systemu lub nie. Konkretne rozwiązania mogą oczywiście zależeć od aplikacji, w tym m.in. od konkretnych definicji obiektów i zdarzeń.

ROMUALD SYNAK, ELŻBIETA GOMULSKA, WŁODZIMIERZ LIPIŃSKI, TOMASZ LIS,
MIROSŁAW OWCZAREK, MARCIN PAWELCZAK, WOJCIECH WIŚNIEWSKI
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Transmisometr laserowy TL 01 - budowa i wyniki badań
Laser transmissometer model TL 01 - construction and results of investigations

Streszczenie

W pracy opisano transmisometr - urządzenie do oceny widzialności horyzontalnej w powietrzu metodą pomiaru transmisji ośrodka. Zastosowano w nim laser diodowy, którego wiązka światła jest podawana do reflektora zwrotnego umieszczonego na wspólnej ramie ze zintegrowanym zespołem nadawczo - odbiorczym. Przyrząd pracuje automatycznie, a wyniki pomiarów są przekazywane do komputera.

Opis transmisometru i wyników jego badań poprzedzono podaniem fizycznych podstaw działania przyrządu i analizą wpływu dokładności pomiaru transmisji na błąd oceny widzialności.

Abstract

This paper describes the transmissometer - an instrument designed to evaluate horizontal visibility in air by measuring the transmittance of the medium. It uses the laser diode whose light beam is derected to the reflector situated together with the transmitter - receiver assembly. The instrument is automatic and sends the measurement results to the computer.

The introduction contains the physical principles on which this instrument operates, and analyses the influence of transparency measurement accuracy on the error of the visibility evaluation.

1. Wstęp

Transmisometry, obok przyrządów opartych na pomiarze światła rozproszonego, stanowią podstawową grupę urządzeń służących do oceny widzialności w powietrzu [1]. Dokonują one pomiaru transmitancji (transmisji) atmosfery wzdłuż linii bazowej prowadzącej od nadajnika do odbiornika światła. Znajomość tego parametru umożliwia obliczenie zasięgu widzialności horyzontalnej definiowanego zgodnie z teorią Koschmiedera jako dystans, na którym kontrast między ciałem czarnym i tłem (horyzontem) spada do progowej czułości oka ludzkiego [2].

Zasięg widzialności oblicza się przy założeniu, że ośrodek jest jednorodny w całym rozpatrywanym obszarze. Wówczas moc promieniowania na podstawie prawa

Bouguera - Lamberta maleje wykładniczo w funkcji odległości od źródła światła ze stałą - współczynnikiem ekstynkcji, który wyraża stopień rozpraszania i absorpcji światła w atmosferze. Ponieważ współczynnik ekstynkcji jest funkcją długości fali światła [3], powyższa zależność zachodzi dla światła monochromatycznego lub w przybliżeniu dla światła białego, gdy w obrębie jego widma zmiany tego współczynnika są niewielkie (ma to miejsce np. przy występowaniu mgły). W tych przypadkach zasięg widzialności horyzontalnej jest odwrotnie proporcjonalny do logarytmu naturalnego mierzonej transmisji i wprost proporcjonalny do długości linii bazowej. Bliższe omówienie fizycznych podstaw działania transmisometru zawarto w artykule [4]. Omówiono w nim również opracowany w Instytucie Maszyn Matematycznych model transmisometru, w którym jako źródło światła zastosowano laser diodowy emitujący światło czerwone. Stwierdzono, że w stosunku do konwencjonalnych urządzeń z lampą ksenonową zastosowanie lasera diodowego daje wiele korzyści: możliwy jest pomiar transmisji ze znacznie wyższą dokładnością, co pozwala na skrócenie długości linii bazowej i zwiększenie zakresu pomiarowego, a konstrukcja transmisometru staje się bardziej zwarta.

Zastosowanie innych źródeł niż światła białego miało już miejsce uprzednio. Wykorzystywano mianowicie lasery He-Ne [5], [6] lub diody LED pracujące w podczerwieni (przyrządy z tymi drugimi elementami przeszły pomyślnie badania porównawcze opisane w sprawozdaniu [1]). Ponieważ jednak definicja meteorologicznego zasięgu optycznego MOR (Meteorological Optical Range) ustanowiona przez Międzynarodową Organizację Meteorologiczną w roku 1957 mówi o użyciu światła lampy żarowej o temperaturze barwowej 2700 K, zapewniającej uzyskanie widma zbliżonego do słonecznego, dokonano analizy, jakie różnice w ocenie widzialności w stosunku do pomiaru zgodnego z definicją MOR mogą powstać przy zastosowaniu światła monochromatycznego [7]. Wykazała ona, że gdy długość fali światła wynosi 550 nm (tj. odpowiada maksymalnej czułości oka ludzkiego przy widzeniu dziennym), różnice są pomijalne dla praktyki (mniejsze niż 2%), a przy innych długościach fal z zakresu pasma widzialnego różniących się od 550 nm można wprowadzić odpowiednią korektę wyniku przez zastosowanie podanej w artykule procedury.

Duże znaczenie pomiaru widzialności dla potrzeb komunikacji lotniczej i lądowej przy braku produkcji krajowej transmisometrów i wysokim koszcie urządzeń zagranicznych, a także zachęcające wyniki uzyskane przy budowie modelu, spowodowały, że postanowiono kontynuować w IMM prace nad transmisometrem. Ich efektem jest opracowanie i wykonanie prototypu transmisometru laserowego, przeznaczonego do pracy w naturalnych warunkach środowiskowych w klimacie umiarkowanym. Opis tego urządzenia, oznaczonego jako TL 01 i wyniki jego badań są przedmiotem niniejszego artykułu.

2. Zasada działania transmisometru

W transmisometrze, niezależnie od wariantu jego budowy, można wyróżnić źródło światła (nadajnik) i umieszczony w pewnej odległości od niego detektor światła (odbiornik). Jeżeli drogę, którą przebywa światło oznaczymy przez x, to przezroczystość atmosfery na tym odcinku może być przedstawiona za pomocą funkcji przepuszczania T(x), będącej stosunkiem strumienia światła mierzonego przez detektor w sytuacji, gdy występuje tłumienie światła w ośrodku, do strumienia mierzonego

w warunkach braku takiego tłumienia. Ze względu na to, że rozpraszanie i absorpcja światła są zależne od długości fali światła λ, a ponadto emisja i detekcja światła ma również charakter spektralny, wyrażenie na funkcję przenoszenia ma w ogólnym przypadku postać całkową. Można ją napisać dla ośrodka jednorodnego przyjmując zgodnie z prawem Bouguera - Lamberta, że moc promieniowania jest funkcją wykładniczą odległości od źródła światła i współczynnika ekstynkcji $\alpha(\lambda)$. Oznaczając przez $P(\lambda)$ i $S(\lambda)$ funkcje spektralne mocy promieniowania źródła i detektora otrzymamy następujące wyrażenie na funkcję $T(x)$:

$$T(x) = \frac{\int P(\lambda)\, S(\lambda) \exp[-\alpha(\lambda)\, x]\, d\lambda}{\int P(\lambda)\, S(\lambda)\, d\lambda}. \qquad (1)$$

Podobną postać do powyższego wzoru ma również wyrażenie na kontrast $C(x)$ ciała czarnego względem tła (zamiast $P(\lambda)$ i $S(\lambda)$ występują w nim radiancja tła i czułość fotometru). Oba te wyrażenia całkowe można uprościć w następujących przypadkach:

1) ograniczając się do pasma widzialnego i przyjmując, że współczynnik ekstynkcji nie ulega w nim dużej zmianie. Założenie takie można dopuścić w sytuacji występowania mgły; na nim opiera się również definicja MOR.
2) przy pomiarze transmitancji atmosfery z użyciem światła laserowego lub innego o wąskim widmie.

Dla wymienionych przypadków otrzymujemy:

$$T(x) = C(x) = \exp(-\alpha x), \qquad (2)$$

gdzie α oznacza średnią wartość współczynnika ekstynkcji w przypadku 1) lub wartość określoną dla długości fali światła lasera w przypadku 2). Z powyższego wzoru można obliczyć zasięg widzialności horyzontalnej V korzystając z definicji Koschmiedera przez przyrównanie wyrażenia do wartości progowej czułości oka ludzkiego ε. Wówczas otrzymuje się:

$$V = -\ln\varepsilon / \alpha. \qquad (3)$$

W transmisometrze mierzy się wartość funkcji $T(x)$ dla linii bazowej o długości B jako:

$$T(B) = T = P(B) / P(0) = \exp(-\alpha B), \qquad (4)$$

gdzie P(B) oznacza moc promieniowania mierzoną przez odbiornik, gdy istnieje tłumienie światła w powietrzu, a P(0) mierzoną moc przy idealnie czystym powietrzu. Ze wzoru (4) można obliczyć współczynnik ekstynkcji:

$$\alpha = -\ln T / B. \qquad (5)$$

Podstawiając wielkość tę do wzoru (3) otrzymuje się:

$$V = B \ln\varepsilon / \ln T. \qquad (6)$$

Do obliczeń V przyjmuje się zwykle wartość ε z przedziału 0,02 - 0,05, przy czym w definicji MOR przyjęto 0,05. Również taką wielkość użyto do wyznaczania widzialności w opisywanym transmisometrze. Wzór (6) upraszcza się wtedy do postaci:

$$V = -3B / \ln T. \qquad (7)$$

3. Analiza wpływu błędu pomiaru transmitancji na oszacowanie widzialności

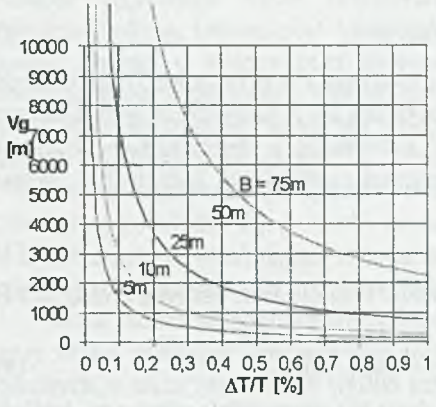
Z przedstawionych uprzednio wzorów widać, że dokładność oceny widzialności zależy od dokładności pomiaru transmisji. Szczególnie krytyczny jest ten wpływ, gdy przeźroczystość powietrza jest bardzo duża i wartość lnT staje się bardzo mała, a także przy bardzo dużym tłumieniu światła, gdy dużą rolę zaczyna odgrywać błąd względny pomiaru T. Aby dokonać analizy wielkości tych wpływów zróżniczkujmy wyrażenie (7) i przejdźmy do przyrostów skończonych ΔV i ΔT. Otrzymuje się wówczas wzór na błąd względny widzialności wyrażony w funkcji błędu względnego transmisji:

$$\Delta V / V = -(1 / \ln T) \cdot \Delta T / T \qquad (8)$$

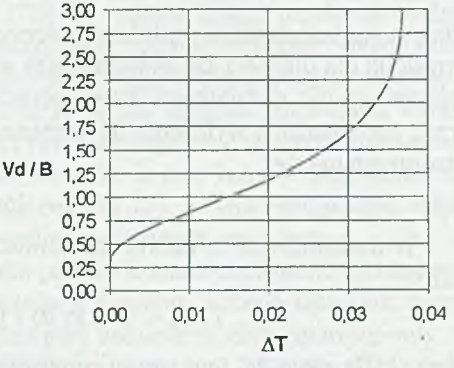
lub bezwzględnego

$$\Delta V / V = -(1 / T \ln T) \cdot \Delta T. \qquad (9)$$

Powyższe wzory umożliwiają obliczenie zakresu mierzonych widzialności w funkcji błędu pomiaru transmisji przy założeniu, że dopuszczalny błąd oceny widzialności nie przekracza pewnej założonej wielkości. Tak więc korzystając z zależności (7) i (8) można obliczyć górną wartość tego zakresu, a z (7) i (9) dolną. Wyniki obliczeń zobrazowano w formie wykresów na rys. 1 i 2 przy założeniu, że błąd oceny widzialności wynosi 10 %.



Rys. 1. Górna granica V dla $\Delta V/V = 10$ %



Rys. 2. Dolna granica V dla $\Delta V/V = 10$ %

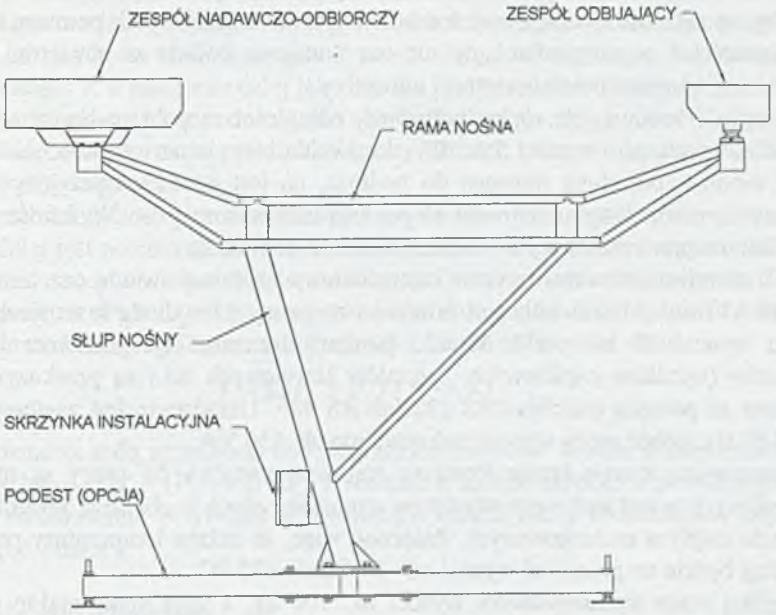
Parametrem jest odległość bazowa B. Jak widać w przypadku transmisometrów z krótką bazą ocena widzialności wyższych niż np. 1000 m wymaga zapewnienia wysokich dokładności pomiaru transmisji - na poziomie kilku dziesiątych części procenta. Ocena widzialności rzędu 10 m przy zastosowaniu takiego transmisometru jest możliwa już przy łagodniejszych wymaganiach.

4. Ogólna struktura urządzenia i jego główne parametry

Podstawowym założeniem przy podjęciu opracowania urządzenia do oceny zasięgu widzialności horyzontalnej było zapewnienie dokładnego pomiaru stanów atmosfery charakteryzujących się występowaniem gęstych mgieł, a jednocześnie umożliwienie

oceny tego parametru podczas zamgleń lub w warunkach zanieczyszczeń przemysłowych. Przyjęto zatem rozwiązanie charakteryzujące się krótką linią bazową i zastosowaniem reflektora zwrotnego, który odbija wiązkę światła i kieruje ją do detektora znajdującego się w sąsiedztwie nadajnika. Uzyskano w ten sposób zwartą i lekką konstrukcję urządzenia, którą można łatwo zamocować na fundamencie lub przenośnym podeście.

Budowę ogólną transmisometru pokazano na poniższym rysunku. Zespoły toru pomiarowego są przymocowane do ramy nośnej - z lewej strony znajduje się zespół nadawczo-odbiorczy, a z prawej odbijający. Odpowiednią wysokość toru pomiarowego uzyskano przez umocowanie ramy na słupie nośnym, którego podstawa jest przykręcana do fundamentu. W dolnej części słupa jest zamocowana skrzynka instalacyjna służąca do dołączenia kabli zasilających i interfejsu. Można do niej podłączyć również sterownik umożliwiający sprawdzenie niektórych parametrów transmisometru oraz wyników pomiaru bez korzystania z komputera.



Rys. 3. Główne zespoły transmisometru TL 01

Zespół nadawczo - odbiorczy zawiera układ pomiarowy, układ mikroprocesorowy, układy elektroniczne sterowania i regulacji temperatury oraz zasilacz. Układy te są umieszczone w zamkniętej i termicznie izolowanej obudowie. Zespół wysyła impulsy światła laserowego, które są kierowane do zespołu odbijającego, skąd wracają do układu pomiarowego. Układ ten mierzy moc promieniowania wiązki światła, co umożliwia obliczenie przez układ mikroprocesorowy tłumienia światła, jakie nastąpiło w wyniku jego przejścia między obu zespołami. Pozwala to z kolei na obliczenie zasięgu widzialności. Dane pomiarowe są przesyłane za pomocą interfejsu do komputera, który dokonuje ich obróbki i rejestracji. Pomiar odbywa się tylko w określonych odcinkach czasu, w których otwierane są migawki znajdujące się w obu zespołach. Do

ich sterowania oraz do regulacji temperatury wewnątrz zespołów służy układ sterowania i regulacji.

Zespół odbijający zawiera pryzmat trójścienny, migawkę oraz elementy grzejne. Podzespoły te są umieszczone w obudowie termoizolacyjnej. Zespół jest umocowany do ramy nośnej za pośrednictwem przesuwnego wspornika, dzięki czemu można odpowiednio wyregulować wzajemne usytuowanie obu zespołów.

Połączenia elektryczne między zespołami są zrealizowane za pomocą kabli i przewodów umieszczonych wewnątrz słupa i ramy. Zaopatrzone są one w złącza umożliwiające łatwe składanie i demontaż urządzenia.

W celu uzyskania wysokiej dokładności pomiaru transmisji, zmierzona moc wiązki światła przychodzącego jest odnoszona do mocy wewnętrznej wiązki odniesienia. Dzięki temu są redukowane wpływy zmian mocy promieniowania lasera, a także wzmocnienia elektronicznego toru pomiarowego. Dalszą poprawę wyników uzyskuje się przez stabilizację temperatury w zespole optycznym transmisometru.

Straty energii świetlnej na elementach optycznych powodują spadek mierzonej transmisji, urządzenie wymaga więc kalibracji sprowadzającej wynik pomiaru transmisji do wartości 1 w przypadku, gdy nie ma tłumienia światła w powietrzu, a więc w warunkach wysokiej przeźroczystości atmosfery.

Odległość bazowa jest równa podwójnej odległości zespołu nadawczo - odbiorczego od odbijającego i wynosi 5 m. Wiązka światła biegnie na wysokości ok. 2 m od miejsca montowania słupa nośnego do podłoża, co jest zgodne z przyjętą praktyką dokonywania pomiarów widzialności za pomocą transmisometrów. Wysokość ta może być zmieniana przez odpowiednie ukształtowanie fundamentu.

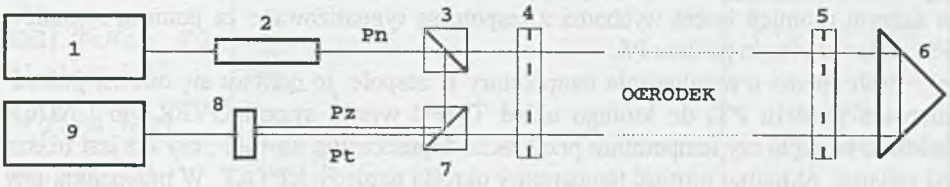
W transmisometrze zastosowano laser diodowy emitujący światło czerwone o długości fali 670 nm. Moc światła jest mierzona za pomocą fotodiody krzemowej. Okres pomiaru wynosi 30 lub 60 s. Wyniki pomiaru transmitancji powietrza i innych parametrów (wyników częściowych, sygnałów alarmowych itd.) są przekazywane do komputera za pomocą interfejsu RS 232 lub RS 485. Urządzenie jest zasilane z sieci 220 V / 50 Hz, pobór mocy wynosi maksymalnie ok. 150 VA.

Opracowana wersja transmisometru jest przeznaczona do pracy w miejscach niechronionych przed wpływem warunków atmosferycznych w obszarze geograficznym o klimacie ciepłym umiarkowanym. Założono więc, że zakres temperatury powietrza, przy której będzie on pracować wynosi od - 25 ^{o}C do +35 ^{o}C.

Ogólna masa transmisometru wynosi ok. 100 kg, a jego maksymalne wymiary gabarytowe są następujące: wysokość 2330 mm, długość 3500 mm, szerokość 280 mm.

5. Układ pomiarowy

Urządzenie mierzy tłumienie światła na odcinku bazowym B równym podwójnej odległości między zespołem nadawczo - odbiorczym i zespołem odbijającym. Pomiaru dokonuje się za pomocą układu złożonego ze źródła światła laserowego, fotodiody oraz z elementów optycznych i mechanicznych. Schemat ogólny toru pomiarowego pokazano na rys. 4.



Rys. 4. Tor pomiarowy transmisometru

Źródło światła laserowego 2 składa się z lasera diodowego, układu kolimującego i układu zasilającego. Jest ono sterowane przez układ 1, który wyznacza czas emitowania światła. Wysyłana jest wiązka skolimowana o mocy promieniowania P_n, która następnie jest rozdzielana w układzie światłodzielącym 3 na wiązkę przechodzącą i wiązkę odbitą. Pierwsza z nich rozchodzi się w badanym ośrodku i po odbiciu od pryzmatu trójściennego 6 wraca do układu padając na drugi układ światłodzielący 7, a następnie dalej jako wiązka P_t dochodzi do fotodiody krzemowej 8. Druga wiązka również jest podawana do układu 7, skąd po odbiciu jako wiązka P_z pada na fotodiodę. Światło może przechodzić przez ośrodek tylko w określonych chwilach, gdy są otwarte migawki 4 i 5.

Sygnał z fotodiody jest wzmacniany i przetwarzany na sygnał cyfrowy w układzie 9, skąd dalej jest podawany do układu mikroprocesorowego transmisometru. Mierzona jest wartość mocy promieniowania zarówno przy migawkach zamkniętych (P_Z), jak i przy otwartych ($P_O = P_Z + P_t$), co pozwala na skompensowanie wpływu zmian mocy promieniowania lasera. Transmitancję ośrodka można obliczyć ze wzoru:

$$T = k(P_o - P_z) / P_z = kT_p \qquad (10)$$

gdzie k oznacza stałą urządzenia uwzględniającą tłumienie światła w elementach optycznych układu, a $T_p = (P_O - P_Z) / P_Z$. Wartość k można określić z powyższego wzoru podczas wzorcowania przyrządu w warunkach bardzo dużej widzialności, gdy transmisja T jest bliska 1.

6. Układy elektroniczne

Schemat blokowy układów elektronicznych transmisometru pokazano na rys. 5. Podstawowe układy elektroniczne urządzenia są zmontowane na następujących pakietach: odbiornika PO, mikroprocesora PM, sterowania PS, regulacji temperatury PT i zasilacza PZ. Pierwszy z nich jest umieszczony w zespole optycznym ZOPT, a pozostałe w oddzielnej kasecie. W transmisometrze znajdują się ponadto następujące płytki z układami elektronicznymi: sterowania silnika migawek PST, ogrzewania migawek UOM, zespołu zasilacza PZZ. Znajdują się one w zespołach, z którymi bezpośrednio współpracują.

W **zespole ZOPT** oprócz pakietu PO znajdują się: laser, fotodioda, dwa sześciany światłodzielące oraz płytka TMPU z układem służącym do pomiaru temperatury i nastawiania przedziału dopuszczalnych jej zmian. Do zmiany temperatury w tym układzie służą elementy grzejne i element chłodzący Peltiera. Do zespołu ZOPT przychodzą

z pakietu PS sygnały: ZLAS sterujący laserem i STS sterujący silnikiem. Okresy czasu, w którym promień lasera wychodzi z zespołu są sygnalizowane za pomocą sygnałów FS, podawanych do pakietu PS.

Jeśli chodzi o regulowanie temperatury w zespole, to odbywa się ono za pośrednictwem **pakietu PT,** do którego układ TMPU wysyła sygnał OVER lub UNDER zależnie od tego, czy temperatura przekracza dopuszczalną wartość , czy też jest niższa od zadanej. Aktualną wartość temperatury określa napięcie VPTAT. W przypadku, gdy temperatura jest za wysoka, pakiet PT wysyła sygnał UTH sterujący elementem Peltiera, a gdy jest za niska, napięcie URg zasilające elementy grzejne.

Do **pakietu PO** przekazywane jest napięcie powstające na fotodiodzie w wyniku padania na nią światła laserowego. Sygnał ten, proporcjonalny do mocy promieniowania, jest wzmacniany i przetwarzany do postaci cyfrowej za pomocą 14 bitowego przetwornika a/c, a następnie przesyłany do pakietu PM. Współpracę między pakietami PO i PM zapewniają następujące sygnały:

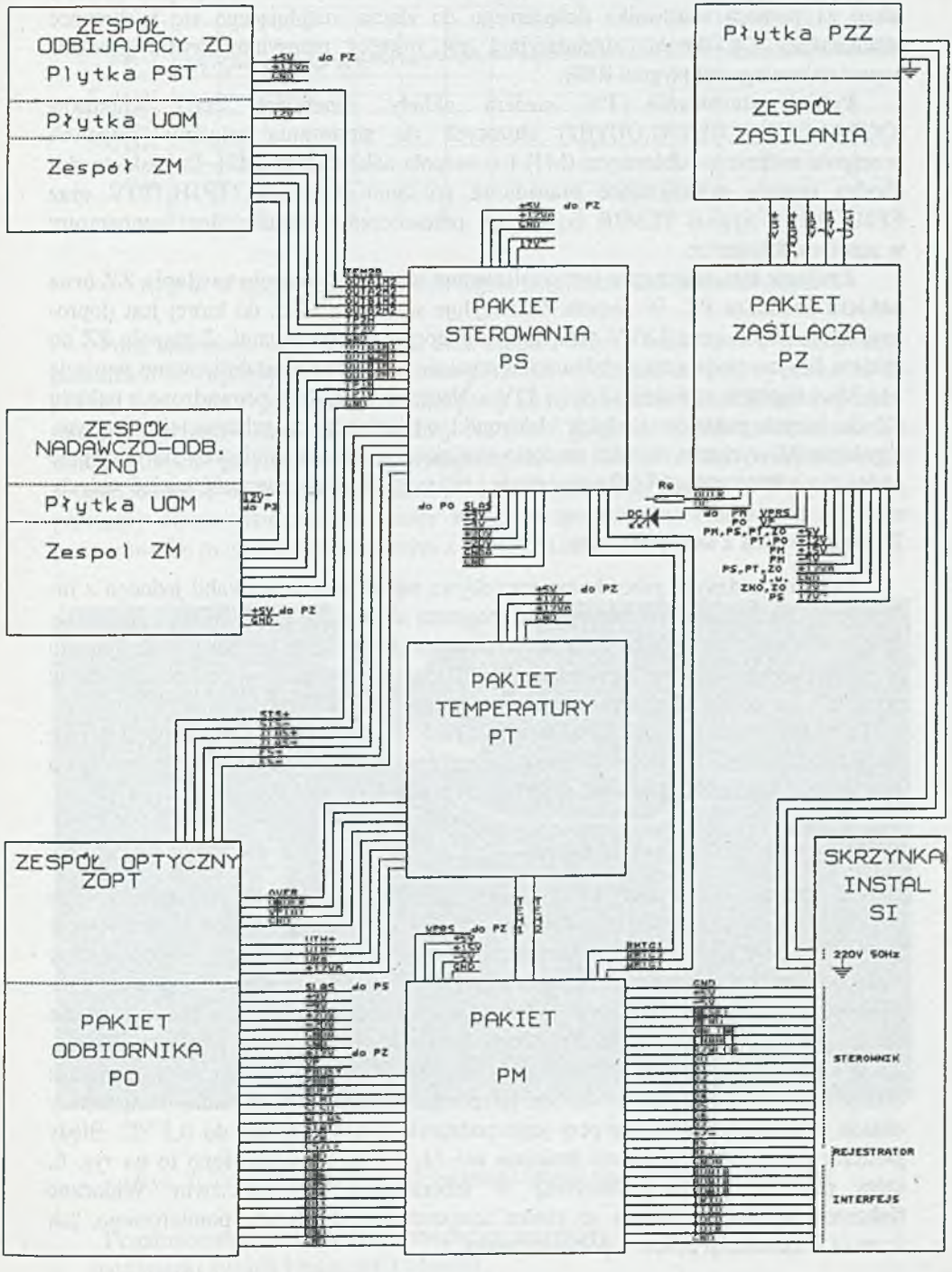
WLLAS - włączanie/wyłączanie lasera,
SLSU - synchronizacja taktowania lasera (stan świecenia/gaszenia),
SLHI - synchronizacja taktowania lasera (stan świecenia),
BUFR - strob odczytu rejestru danych pomiarowych,
PHBN - wybór do odczytu młodszego/starszego bajtu przetwornika a/c,
PBUSY - sygnalizacja czasu przetwarzania przetwornika j.w.,
STAT - gotowość odczytu z przetwornika a/c pomiaru temperatury,
R/W - odczyt sygnału z przetwornika j.w.
CS/CE - strob odczytu danych pomiarowych temperatury.

Pakiet mikroprocesora PM rejestruje dane przekazane z pakietu PO i oblicza wartości średnie mierzonych parametów dla każdego cyklu pomiarowego na podstawie 64 odczytów mocy światła z uwzględnieniem tła. Tak uzyskane wielkości P_Z i P_O, a także obliczone na ich podstawie wartości transmisji i zakresu widzialności są przekazywane do komputera za pomocą interfejsu RS 232 lub RS 485. Pakiet PM pełni ponadto funkcje sterowania podzespołami elektrycznymi i rejestruje ich stany.

Związane są z tym następujące sygnały:

TEM1 - przekroczenie zadanej temperatury lasera,
TEM2 - brak właściwej temperatury w zespole odbijającym lub zespole optycznym,
WMIG1 - włączenie/wyłączenie migawek,
RMIG2 - sygnalizacja stanu otwarcia migawek,
VPRS - awaria zasilania +12V.

Informacje o stanach awaryjnych lub braku gotowości są przesyłane również do komputera.



Rys. 5. Schemat blokowy elektroniki transmisometru.

Oprócz wyjścia interfejsowego dostęp do układu mikroprocesorowego jest możliwy także za pomocą sterownika dołączanego do złącza znajdującego się w skrzynce instalacyjnej. Do skrzynki instalacyjnej jest również przesyłany wynik pomiaru w postaci analogowej (sygnał REJ).

Pakiet sterowania PS zawiera układy generujące ciągi impulsów (OUTA1,OUTA2,OUTB1,OUTB2) służących do sterowania silników migawek w zespole nadawczo-odbiorczym (M1) i w zespole odbijającym (M2). Do pakietu dochodzą sygnały sygnalizujące prawidłowe położenie migawek (TP1H,TP1V oraz TP2H,TP2V). Sygnał TEM2B sygnalizuje przekroczenie dopuszczalnej temperatury w zespole odbijającym.

Zasilanie transmisometru jest zrealizowane za pomocą **zespołu zasilania ZZ oraz pakietu zasilacza PZ.** W zespole ZZ znajduje się płytka PZZ, do której jest doprowadzane napięcie sieci 220 V oraz transformatory z prostownikami. Z zespołu ZZ do pakietu PZ jest podawane stabilizowane napięcie +5V oraz niestabilizowane napięcie +12 Vn i napięcia zmienne 12 V~ i 17V~. Napięcia te są dalej prowadzone z pakietu PZ do innych pakietów i płytek elektroniki wskazanych na schemacie blokowym. Z pakietu PZ wychodzi również napięcie zasilające grzałek znajdujących się w zespole nadawczo-odbiorczym (OUTR) oraz diody LED sygnalizującej obecność napięć stałych.

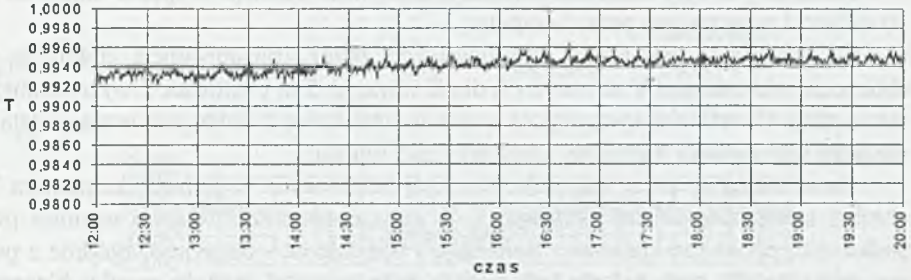
7. Współpraca z komputerem

Rejestracja danych przez komputer odbywa się po dołączeniu kabli jednego z interfejsów i uruchomieniu opracowanego programu obsługi TR.EXE. Dane są automatyczne wyświetlane na ekranie i zapamiętywane na dysku aż do naciśnięcia dowolnego przycisku klawiatury. W zbiorach danych na dysku jest umieszczana informacja o dacie pomiarów uzyskanej na podstawie czasu systemowego komputera.

Użytkownik może napisać własny program obsługi wykorzystujący rejestrowane przez transmisometr dane. Przesłanie do komputera określonych danych odbywa się po przesłaniu do urządzenia odpowiadających im poleceń.

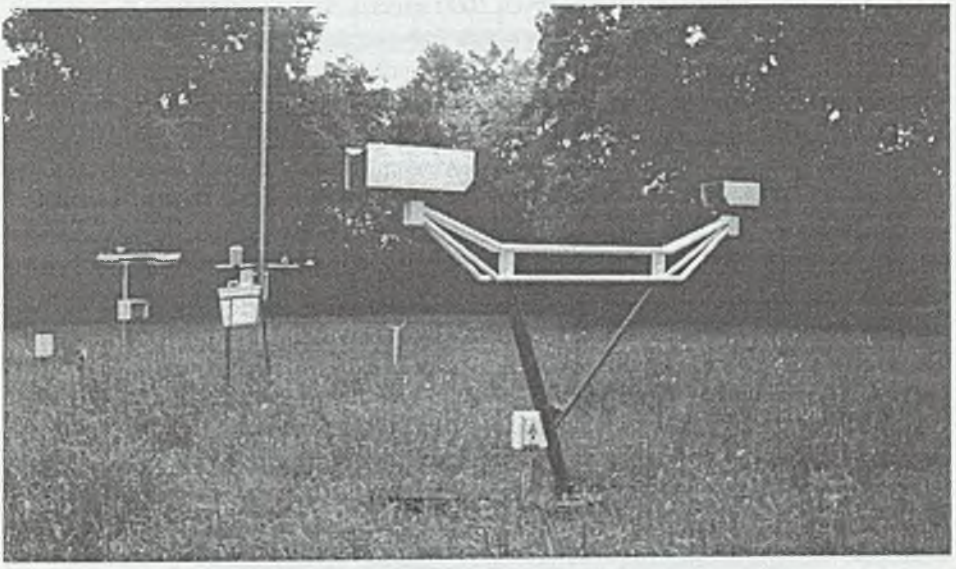
8. Wyniki badań transmisometru

W trakcie prac nad transmisometrem stwierdzono, że warunkiem jego prawidłowego funkcjonowania jest zapewnienie wysokiej stabilności temperatury lasera i układów optycznych. Potrzeba taka wynika stąd, że mimo redukcji wpływu zmiany mocy promieniowania lasera przez zastosowanie układu odniesienia, przy zmianie temperatury powstają błędy pomiaru transmisji spowodowane zmianą tej mocy w czasie różniącym właściwy pomiar od pomiaru mocy wiązki odniesienia, a także zmianą rozkładu mocy promieniowania w wiązce światła. Jak wykazały badania transmisometru zastosowany w nim układ stabilizacji temperatury zapewnia utrzymanie temperatury układu optycznego mierzonej przy jego podstawie z dokładnością do 0,5 ^{O}C. Błędy pomiaru transmisji są wówczas mniejsze niż +/- 0,1 %. Przedstawiono to na rys. 6, który pokazuje przebieg transmisji w laboratorium elektronicznym. Widoczne fluktuacje wynikają zarówno ze zmian temperaturowych układu pomiarowego, jak i zmian koncentracji pyłów w pomieszczeniu.



Rys. 6. Względna zmiana transmisji w pomieszczeniu laboratoryjnym

Poza badaniami laboratoryjnymi oraz krótkotrwałymi badaniami w niskich temperaturach zewnętrznych, transmisometr poddano w okresie od połowy kwietnia do połowy lipca 1996 r. próbie długotrwałej pracy na terenie stacji meteorologicznej Instytutu Meteorologii i Gospodarki Wodnej. Na rys. 7 pokazano usytuowanie urządzenia, wśród innych przyrządów meteorologicznych. Z lewej strony transmisometru w odległości 12,5 m widać przyrząd do pomiaru widzialności typ FD 12 firmy Vaisala, pracujący na zasadzie pomiaru mocy światła rozproszonego. Porównanie wyników pomiarów obu przyrządów było jednym z głównych celów badań.



Rys. 7. Transmisometr TL 01 na terenie stacji meteorologicznej IMGW

Przedmiotem pomiarów były następujące parametry:
- moc wiązki światła i transmisja ośrodka,
- wielkości charakteryzujące działanie przyrządu (temperatura zespołu optycznego, poziom tła).

Parametry te były mierzone automatycznie przez przyrząd w sposób ciągły co ok. 10 sekund i rejestrowane przez komputer.

Na podstawie pomiarów wykonywanych przez transmisometr dokonuje się obliczenia widzialności V ze wzoru (7) dla B równego 5 m i wartości T wyznaczonej ze wzoru (10). Określenia występującej w tym wzorze stałej k dokonywano na podstawie wskazań widzialności mierzonej przez przyrząd Vaisali.

Dane mierzone przez transmisometr były zapisywane w pamięci komputera bez obróbki selekcjonującej lub korygującej. W celu zmniejszenia wpływu wartości przypadkowych parametry końcowe - transmisję i widzialność - obliczano, zgodnie z przyjętą praktyką dla tego rodzaju przyrządów, jako wartości średnie wyniku bieżącego i poprzedniego. Nie stosowano natomiast bardziej złożonej obróbki, która eliminowałaby wyniki całkowicie błędne lub obarczone dużymi zakłóceniami.

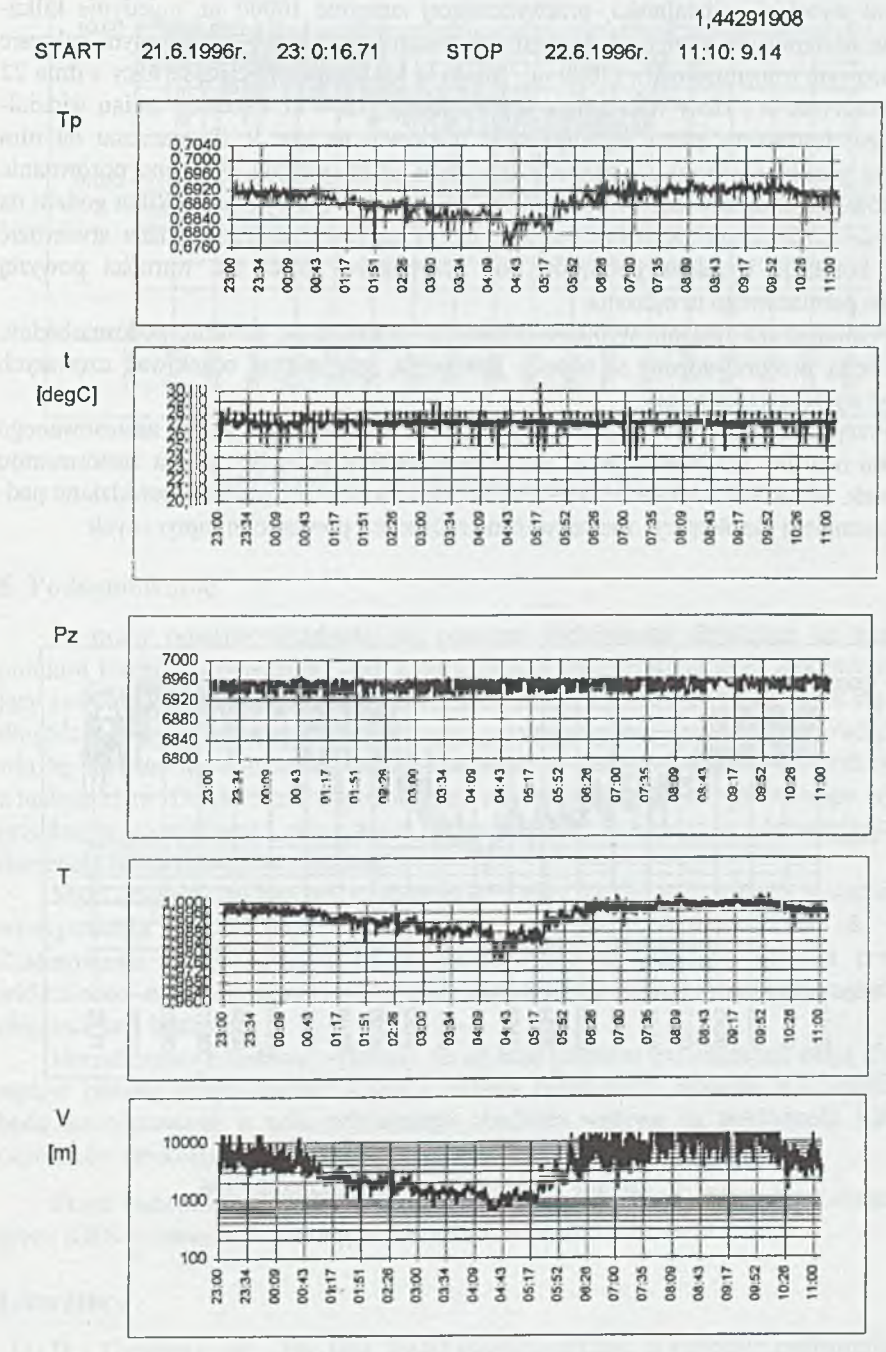
Instytut Meteorologii i Gospodarki Wodnej udostępnił wyniki pomiarów parametrów meteorologicznych mierzonych na stacji, w tym zakresu widzialności. Obejmowały one wartości chwilowe mierzone o pełnej godzinie oraz wartości średnie dla okresu 10 minut poprzedzających odczyt wartości chwilowej.

Dane meteorologiczne zostały poddane dalszej obróbce i analizie, co pozwoliło na wyodrębnienie okresów, w których występowało obniżenie widzialności lub też mogło mieć miejsce zwiększone oddziaływanie niektórych czynników zewnętrznych na pracę urządzenia.

Wyniki pomiarów transmisometru były pamiętane w pamięci dyskowej komputera w formie zbiorów rekordów obejmujących 1000 wierszy danych zebranych w czasie ok. 3 godzin. Oprócz danych za każdym razem były zapisywane: symbol rekordu, data pomiaru oraz czas rozpoczęcia i zakończenia pomiarów tego rekordu, a ponadto wprowadzane podczas inicjowania pracy urządzenia dane wyznaczające stałą k.

Dane pomiarowe poddano dalszej obróbce przy wykorzystaniu programu Microsoft EXCEL 4.0. Kolumny z danymi pomiarowymi zostały uzupełnione o kolumny zawierające obliczone dla określonej wartości współczynnika k wielkości transmisji T i odpowiadającej jej widzialności V. Oprócz kolumn zawierających nieobrobione wartości tych parametrów na arkuszu zamieszczono również wartości średnie. Na podstawie tych danych wytworzono wykresy zmian niektórych parametrów mierzonych przez transmisometr, a także transmisji i widzialności. Wykresy pokazujące przebiegi dla okresu 12 godzinnego zgromadzono w celu łatwiejszego porównania na jednym arkuszu. Przykładowy wygląd arkusza pokazano na rys. 8.

Opisane arkusze wykonano dla całego okresu badań. Pozwoliło to na skorelowanie uzyskanych wyników z pomiarami parametrów meteorologicznych dla interesujących sytuacji pogodowych, a w szczególności z pomiarami widzialności dokonywanymi przez urządzenie FD 12. Dla takich przypadków sporządzano dodatkowe wykresy. Szczegółowe wyniki badań przedstawiono w opracowaniu [8], tutaj przedstawimy jedynie niektóre rezultaty odnoszące się do pomiarów widzialności.

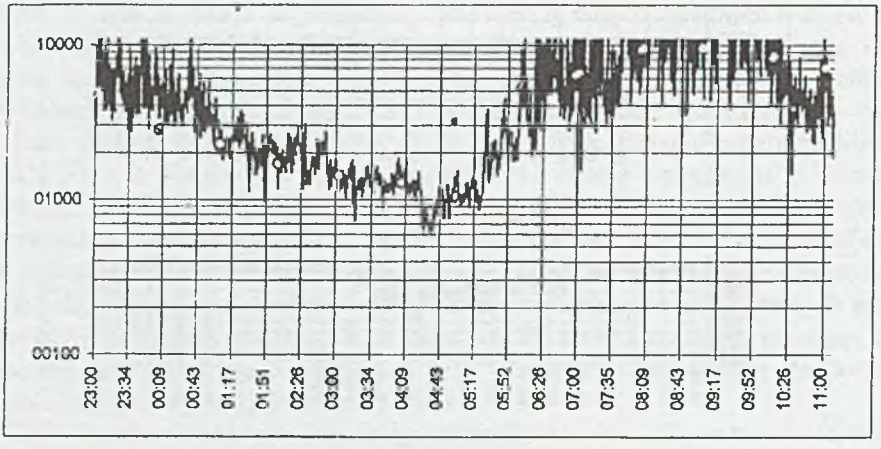


Rys. 8. Wygląd arkusza zawierającego przebiegi zmian podstawowych parametrów transmiometru

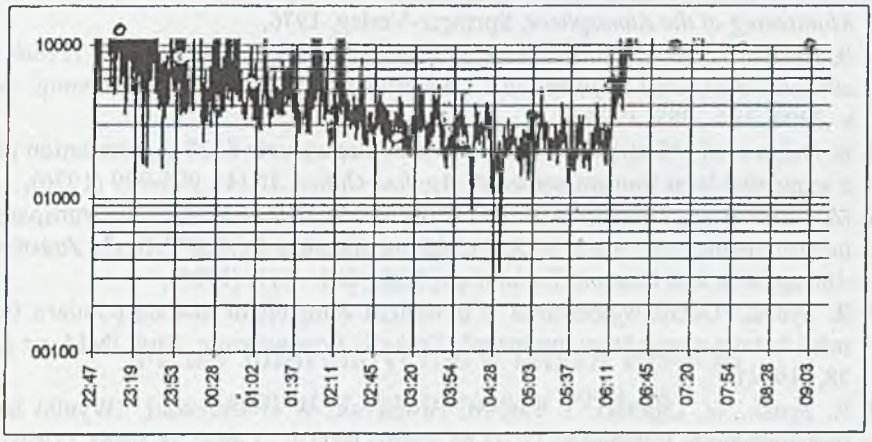
Okres, w którym przeprowadzono badania charakteryzował się ogólnie biorąc stanami wysokiej widzialności, przewyższającej znacznie 10000 m, a jedynie kilkakrotnie nastąpiło jej obniżenie do wartości mieszczących się w zakładanym zakresie pomiarowym transmisometru (3000 m). Miało to szczególnie miejsce w nocy z dnia 22 na 23 czerwca br., kiedy widzialność spadła poniżej 1000 m. Przebieg zmian widzialności zarejestrowany przez transmisometr pokazano na rys. 9. Zaznaczono na nim również wyniki uzyskane za pomocą przyrządu firmy Vaisala. Podobne porównanie wyników uzyskanych w dn. 17/18 maja br., gdy widzialność była przez kilka godzin na poziomie 2000 m, przestawiono na rys. 10. W obu przypadkach można stwierdzić dobrą korelację wyników pomiarów obu przyrządów nawet dla wartości powyżej zakresu pomiarowego urządzenia.

Pełniejsze porównanie wyników pomiarów zamierza się uzyskać podczas badań, które będą przeprowadzone w okresie jesiennym, gdy można oczekiwać częstszych stanów występowania mgieł.

Przeprowadzone badania pozwoliły również na ocenę skuteczności zastosowanego sposobu ochrony urządzenia przed zanieczyszczeniami polegającego na zastosowaniu migawek. Mimo ich trzykrotnie większego czasu otwarcia niż jest to przewidziane podczas normalnej eksploatacji, nie zauważono zabrudzeń powierzchni optycznych.



Rys. 9. Zmiany widzialności w dn. 22/23.06.1996



Rys. 10. Zmiany widzialności w dn. 17/18.05.1996.

6. Podsumowanie

W pracy opisano urządzenie do pomiaru widzialności działające na zasadzie pomiaru transmisji powietrza. Jako źródło światła wykorzystano laser diodowy emitujący światło czerwone o długości fali 670 nm. Transmisometr charakteryzuje się małą długością linii bazowej wynoszącą 5 m i zastosowaniem reflektora zawracającego wiązkę światła do odbiornika znajdującego się w jednym zespole konstrukcyjnym z nadajnikiem. Dzięki temu transmisometr odznacza się małymi, jak na tego rodzaju urządzenia, rozmiarami i masą. Może być instalowany w terenie na odpowiednim fundamencie lub przenośnym podeście.

Mała długość linii bazowej umożliwia dokładny pomiar widzialności w warunkach występowania gęstych mgieł, nawet przy ograniczeniu widzialności do ok. 5 m. Zastosowanie lasera pozwala na uzyskanie znacznie szerszego zakresu pomiaru widzialności niż było to możliwe w transmisometrach konwencjonalnych o podobnej długości linii bazowej.

Dotychczasowe badania wykazały, że na błąd pomiaru transmitancji mają głównie wpływ zmiany temperaturowe lasera i układu optycznego. Badania transmisometru będą kontynuowane w celu pełniejszego zbadania wpływu na dokładność pomiaru czynników środowiskowych oraz stabilności układu pomiarowego.

Prace badawczo - rozwojowe nad transmisometrem były częściowo finansowane przez KBN w ramach projektu nr 8 8524 93 C/1297.

Literatura

[1] D.J. Grigges et al., "The first WMO intercomparison of visibility measurements", *Instruments and Observing Methods*, WMO Report No. 41, 1989,

[2] V.A. Gavrilov, *Visibility in the Atmosphere*, Springfield, VA, 1966,

[3] V.E. Zujev, "3.Laser-light transmission through the atmosphere", *Laser Monitoring of the Atmosphere*, Springer-Verlag, 1976,
[4] R. Synak, "Laser transmissometer", *Laser Technology IV - Research Trends, Instrumentation, and Application in Metrology and Materials Processing*, SPIE v. 2202, 384 - 388, 1995,
[5] B. Daino et al., "Statistical measurement of atmospheric 6328-A attenuation using a sequential laser transmissometer", *Applied Optics*, 15 (4), 996-999 (1976),
[6] M. Gazzi et al., "Diagnosis of the causes of systematic errors in the transparency measurements and some experimental verification in Po Valley", *Journal of Atmospheric and Oceanic Technology*, 2 (6), 201 - 211, (1985),
[7] R. Synak, "Ocena widoczności w powietrzu zamglonym metodą pomiaru transmisji światła monochromatycznego", *Techniki Komputerowe*, Biul. IMM, nr 1, 5 - 28, (1994),
[8] R. Synak, W. Lipiński, T. Lis, M. Pawelczak, W. Wiśniewski, "Wyniki badań transmisometru laserowego TL 01 na terenie IMGW", Oprac. w. IMM, (1996).

JAN RYŻKO
INSTYTUT MASZYN MATEMATYCZNYCH WARSZAWA

Firmy informatyczne w latach 1994-96
Computer companies in 1994-96

Streszczenie

W pracy przedstawiono rankingi firm komputerowych w roku 1995 i zmiany jakie w nich nastąpiły w stosunku do roku 1994 zarówno na świecie (na podstawie Datamation 100 [1]) jak i w Polsce (Raport Specjalny Computerland [2]). W oparciu o te dane wyciągnięto wnioski odnośnie światowej i krajowej kondycji informatyki i tendencji rozwojowych w tej dziedzinie. Przeprowadzono próbę porównania sytuacji w Polsce i na świecie.

Abstract

This article presents the largest computer companies listed by their IT revenues in 1995 and revenue changes from the priror year in Poland and worldwide. Based on this information the author has drawn some conclusions on the trends in the computer industry, and tried to compare the situation in Poland to the rest of the world.

Lista Datamation

Śledząc rozwój światowego rynku komputerowego od połowy lat osiemdziesiątych na podstawie wzrostu przychodów stu największych firm informatycznych na świecie można stwierdzić, że był on bardziej dynamiczny niż cała gospodarka, przy czym w drugiej połowie lat osiemdziesiątych tempo wzrostu malało (od ponad 19% do niespełna 7% rocznie), a w pierwszej połowie lat dziewięćdziesiątych rosło osiągając niemal początkową wartość w roku 1995 (19,1%) [1]. Ostatnio pojawiły się pewne sygnały [3] świadczące o ponownych problemach tego rynku, ale trudno jeszcze ocenić, czy jest to trwała tendencja.

W roku 1995 suma przychodów z dziedziny informatyki firm znajdujących się na liście Datamation 100 wyniosła 439,1 mld USD, przy czym 75 z nich miało przychody wyższe od 1 mld USD (w roku 1994 - 63, 1990 - 52, a w 1985 - 31). Zwraca uwagę to, że większe przychody były (i są) również skutkiem łączenia się firm i nabywania zwykle mniejszych firm przez większe. Tabela 1 przedstawia 30 czołowych firm informatycznych na świecie w 1995 roku.

Tabela 1

Poz. 1995	Poz. 1994	Firma	Kraj	Przychody (mln USD) 1994	1995	Zmiana %	Zysk mln USD	Procent przych. z informatyki
1	1	IBM	USA	64652	71940	12,3	4178	100
2	2	Fujitsu	Japonia	21331	26798	25,6	671	67
3	3	Hewlett-Packard	USA	19200	26073	35,8	2433	80,2
4	4	NEC	Japonia	18726	19350	3,3	700	45
5	5	Hitachi	Japonia	13699	16208	18,3	br. d.	19
6	8	Compaq Computer	USA	10866	14800	36,2	789	100
7	6	Digital Equipment	USA	13500	14440	7	431	100
8	9	Electronic Data Systems	USA	10052	12422	23,6	939	100
9	7	AT&T	USA	11459	11384	-0,7	139	14,3
10	10	Toshiba	Japonia	9936	11380	3,1	850	18,9
11	11	Apple Computer	USA	9549	11378	19,2	167	100
12	12	Siemens Nixdorf	Niemcy	7209	8951	24,2	16	100
13	24	Seagate Technology	USA	3500	8200	134,3	br. d.	100
14	19	Microsoft	USA	4650	7418	59,5	1838	100
15	14	Matsushita	Japonia	6058	7026	16	1017	9
16	17	Sun Microsystems	USA	5348	6500	21,6	447	100
17	13	Unisys	USA	6216	6202	-0,2	-625	100
18	18	Olivetti	Włochy	5303	6035	13,8	br. d.	100
19	26	Acer	USA	3200	5700	78,1	br. d.	100
20	15	Canon	Japonia	5708	5616	-1,6	429	47
21	16	Groupe Bull	Francja	5388	5300	-1,6	61	100
22	23	Dell Computer	USA	3500	5296	51,3	272	99,9
23	22	NTT Data	Japonia	3800	5287	39,1	81	100
24	20	Xerox	USA	3983	4668	17,2	-472	28,1
25	32	Packard Bell	USA	2600	4300	65,4	br. d.	100
26	34	Andersen Consulting	USA	2452	4220	72,1	br. d.	100
27	25	Quantum	USA	3286	4174	27	56	100
28	21	Mitsubishi	Japonia	3848	4145	7,7	br. d.	12
29	27	Computer Sciences	USA	3085	4100	32,9	133	100
30	28	Cap Gemini Sogeti	Francja	2955	3785	28,1	br. d.	90

Analizując tę tabelę, a także pełną listę stu firm łatwo zauważyć, że charakteryzuje się ona dość dużą stabilnością, szczególnie pierwsza jej połowa. Dwanaście firm (na sto) nie zmieniło swej pozycji na liście w stosunku do 1994 roku, w tym aż sześć w pierwszej dziesiątce. Dwadzieścia pięć firm zmieniło tę pozycję tylko o 1-2 miejsca. Nowych firm pojawiło się 15, w tym tylko jedna w pierwszej pięćdziesiątce na miejscu 34. Kolejność pierwszej piątki nie uległa zmianie, aczkolwiek widoczne jest zbliżenie Hewlett-Packarda do Fujitsu oraz Hitachi do NEC. Na czele listy od początku prowadzenia tej statystyki (20 lat) znajduje się IBM z prawie trzykrotną przewagą nad

najbliższym rywalem, aczkolwiek jej tempo wzrostu jest niższe w stosunku do firm będących na 2-3 lub 5-6 miejscu.

Największe tempo wzrostu (188,2%) osiągnęła zajmująca się m.in. modemami firma US Robotics, której nie było na liście Datamation 100 w 1994 roku, a w 1995 roku zajęła 70 miejsce. Jest to przykład szybkiego rozwoju firm telekomunikacyjnych, choć niemały udział ma tu zakup przez US Robotics firmy Megahertz za 6,3 mln swoich akcji w lutym 1995 roku, a potem jeszcze ISDN Systems Corporation i Palm Computing. US Robotics zajmuje 11 miejsce na liście firm o największych przychodach z komunikacji danych (Tabela 7). Drugie miejsce wśród najszybciej rozwijających się firm zajmuje Seagate Technology (134,3 % wzrostu przychodów i awans z 24 na 13 miejsce w Tablicy 1). Jednocześnie Seagate zajmuje trzecie miejsce wśród firm o najwyższych przychodach z urządzeń peryferyjnych (Tabela 4) i jej sukces jest przykładem znacznego wzrostu zapotrzebowania na pamięci dyskowe. Według ostatnich danych [4] pojemność ich wzrasta o 60% rocznie, to znaczy podwaja się co pięć lat. Trzecie miejsce zajmuje firma 3 Com (99,6% wzrostu i awans z 63 na 48 miejsce) jako kolejny przykład rozwoju firm telekomunikacyjnych (3 miejsce w Tabeli 7), a czwarte Acer (78,1% i awans z 26 na 19 miejsce) - tym razem przykład rozwoju sprzedaży komputerów osobistych i stacji roboczych (12 miejsce w Tabeli 3). Pierwszą piątkę najszybciej rozwijających się firm zamyka firma Cisco Systems (77,9% wzrostu i awans z 49 na 38 miejsce) - to już trzecia w tej piątce firma telekomunikacyjna (czwarte miejsce w Tabeli 7). Mimo tak wyraźnego przyspieszenia rozwoju firm telekomunikacyjnych w ostatnim okresie, co niewątpliwie wiąże się z rozwojem sieci komputerowych i Internetu, udział tej dziedziny w globalnych przychodach rynku informatycznego spadł w ostatnim dziesięcioleciu (1985-95) z 8 do 7%. Podobna sytuacja dotyczy serwerów (spadek z 11 do 8%). Również urządzenia peryferyjne odnotowały pewien spadek udziału (z 27 do 19%). Wyraźny natomiast spadek dotyczy dużych systemów (z 17 do 6%). Udział pozostałych dziedzin rynku informatycznego wzrósł: najbardziej komputery osobiste i stacje robocze - z 10 do 22%, następnie usługi - z 17 do 27% i oprogramowanie - z 8 do 11%. Przykładem z tej ostatniej dziedziny jest firma Microsoft, zajmująca 10 miejsce wśród najszybciej rozwijających się firm - 59,5% wzrostu i awans o 5 miejsc w Tabeli 1.

Wracając do listy Datamation 100 największy awans osiągnęła tu firma Cooper & Lybrant działająca w usługach, która przesunęła się w górę o 26 miejsc na miejsce 60, a której nie wymieniono wśród najszybciej rozwijających się firm, gdyż zwiększyła ona jednocześnie udział informatyki w przychodach. Na liście firm o najwyższych dochodach z usług zajmuje ona 22 miejsce. O 18 miejsc na miejsce 59 awansowała firma Kingston Technology zajmująca się głównie urządzeniami peryferyjnymi, ale w tej dziedzinie nie zdołała wejść do 25 najlepszych firm. Natomiast największego spadku (o 22 miejsca na miejsce 89) doznała firma Cray Research. Wiąże się to z wspomnianym ogólnym zmniejszeniem udziału dużych systemów oraz wewnętrznymi trudnościami firmy, co doprowadziło do częściowego jej wykupienia przez Sun Microsystems [5]. Spadki o 18 miejsc odnotowały: na 47 miejsce japońska firma Ricoh najbardziej znana z drukarek, której przychody zmniejszyły się o 31,5%, oraz na 80 miejsce Price Waterhouse, zajmująca się usługami.

W pierwszej dziesiątce mamy sześć firm ze Stanów Zjednoczonych i cztery japońskie. Jako pierwsza spoza tych dwóch państw pojawia się na 12 miejscu firma Siemens Nixdorf z Niemiec, a potem na 18 miejscu włoska Olivetti i na 21 Groupe

Bull z Francji. W całej pierwszej setce jest 69 firm z USA, 14 z Japonii, 5 z Francji, 3 z Tajwanu, po 2 z Niemiec, Włoch i Wielkiej Brytanii oraz po jednej z Korei Pd., Szwecji i Kanady.

Wśród firm pierwszej dziesiątki tylko cztery (IBM, Hewlett-Packard, Digital Equipment i Electronic Data Systems) były notowane na liście Datamation 100 w 1980 roku. Biorąc pod uwagę wzrost przychodów tej dziesiątki na przestrzeni 20 lat, był on najwyższy w pięcioleciu 1975-80 (ponad 4-krotny), w następnych dwóch pięcioleciach przychody wzrastały średnio 3-krotnie w ciągu 5 lat, a w ostatnim pięcioleciu (1990-95) był najniższy, gdyż średnio tylko nieco ponad dwukrotny. Największy ten współczynnik (7,14) osiągnęła firma Compaq w latach 1985-90, najniższy zaś (1,07) - IBM w ostatnim pięcioleciu. Określając analogiczne parametry dla całej setki można stwierdzić, że są one ponad 1,5-krotnie niższe w porównaniu z pierwszą dziesiątką.

Biorąc pod uwagę zysk (dla pełnej listy) na czele znów znajduje się IBM - 4,2 mld USD, potem zajmujący 32 pozycję Intel - 3,6 mld, H-P - 2,4 mld, Microsoft i Motorola - po 1,8 mld. Natomiast największe straty poniosła zajmująca 64 pozycję firma Alcatel - 5 mld USD, następnie Unisys - 625 mln i Xerox - 472 mln.

Inne rankingi światowe

Zanim przejdziemy do omawiania rankingów w poszczególnych dziedzinach informatyki, przytoczmy za [6] krótkie omówienie innej listy, a mianowicie Global 1000 publikowanej przez Business Week i przedstawiających tysiąc największych firm światowych wg wartości giełdowej w dniu 31 maja 1996 roku. Inne są więc kryteria doboru, a jednocześnie dane dotyczą bliższego czasowo okresu i mogą być traktowane jako kontynuacja Datamation 100. W pierwszej dziesiątce tej listy, którą otwiera General Electric z USA z wartością 137,3 mld USD, a która zajmowała dalekie 73 miejsce na liście Datamation, są jeszcze tylko dwie firmy z tej listy, obie telekomunikacyjne, przy czym pierwsza z nich, zajmująca 3 miejsce (115,7 mld USD), NT&T z Japonii jest szerszą organizacją aniżeli NTT Data zajmująca 23 miejsce Datamation 100. Firma ta była pierwszą na liście Global 1000 w 1995 roku. Druga zaś, zajmująca siódme miejsce z wartością niespełna 100 mld USD, która w roku 1995 była piąta na tej liście, a w Datamation 100 dziewiąta, to AT&T. Wśród pierwszych 25 pozycji listy Global znalazły się jeszcze cztery firmy informatyczne z USA: Microsoft na pozycji 12 (20 w 1995) z wartością 71 mld USD, Intel - odpowiednio poz. 15/21 i 62 mld, IBM - 20/18 i 57,6 mld oraz Hewlett-Packard - 23/42 i 54 mld. Tak więc kolejność jest tu inna niż na liście Datamation, co jest konsekwencją przyjętych kryteriów, ale również widać w większości awans firm informatycznych. Największy awans w pierwszej setce Global osiągnęła znana już nam z Datamation firma Cisco Systems, która tu przesunęła się z miejsca 214 na 63 przy wartości 30,5 mld USD. Również w pierwszej setce, na miejscu 90, pojawiła się z wartością 24,2 mld USD oferująca produkty teleinformatyczne firma Lucent Technologies, która powstała z akcji AT&T (stąd spadek tej ostatniej). Wśród innych nowych firm informatycznych na liście Global są również znane nam z Datamation: US Robotics i NTT Data - tu na miejscach 402 i 403 z wartością 7,8 mld USD oraz Seagate Technology - poz. 513 i 6,2 mld i Dell Computer - 639 i 5,15 mld, a także zupełnie nowe: Parametric Technology (oprogramowanie CAD/CAM) - poz. 554 i 5,8 mld USD, Netscape Communications (przeglądarki w Internecie) - 570 i 5,6 mld, Iomega (napędy wymienne) - 634 i 5,2 mld, America

Online (jedyny dostawca Internetu na liście) - 673 i 4,9 mld, SMC (jedyna nowa informatyczna firma japońska) - 685 i 4,8 mld, Fore Systems (produkty sieciowe) - 888 i 3,6 mld, PeopleSoft (aplikacje finansowo-księgowe) - 904 i 3,5 mld i Analog Devices (układy scalone) - 998 i 3,2 mld. Jeśli uszeregowalibyśmy firmy informatyczne z listy Global według wzrostu wartości giełdowej, to na czele znalazłaby się Iomega z prawie 18-krotnym wzrostem, następnie US Robotics ze wzrostem 3,4-krotnym, America Online (217%) i Fore Systems (199%).

Natomiast największy awans wśród firm informatycznych całej listy uzyskała Sun Microsystems, która była 16 na liście Datamation, a tu awansowała z miejsca 644 na 261 osiągając wartość 11,5 mld USD, następnie kanadyjska Newbridge Networks - z 910 na 534 i 6 mld, First Data (usługowe przetwarzanie danych, podwykonawstwo) - z 441 na 149 i 17,8 mld i 3COM - 48 na liście Datamation, tu z 636 na 375 i 8,2 mld. Wśród 54 firm informatycznych na liście Global znalazło się 13 firm nowych, 25 poprawiło swą pozycję, a 16 pogorszyło. Na czele tych ostatnich znajduje się Apple Computer - spadek z 543 na 978 pozycję przy wartości 3,2 mld, następnie Novell - z 393 na 632 i 5,2 mld i Silicon Graphics - z 501 na 725 i 4,5 mld. Spadły też amerykańskie firmy półprzewodnikowe Micron Technology i LSI Logic.

Na liście Global nie ma firm z innych krajów dalekowschodnich. Są one zaliczane do „powstających rynków" i umieszczane na oddzielnej liście Top 200, na której z 7 na 13 miejsce spadła koreańska firma Samsung z wartością 10,2 mld USD - na liście Datamation była ona 57. Dalej mamy tu na 44 pozycji nową firmę Taiwan Semiconductor z wartością 5,3 mld i spadłą z 44 na 77 miejsce również tajwańską UMC z wartością 3,6 mld.

Ostatnim rankingiem z listy Global, o którym powinno się krótko wspomnieć, jest klasyfikacja największych firm o największych zyskach. Nie jest to oczywiście dokładna wartość zysku za cały rok 1996, gdyż dane zebrane były przed upływem połowy tego roku, ale na pewno oddaje tendencje w tym zakresie, jakie obserwowano do końca maja. Tu również istnieje zupełnie inna kolejność. Na pierwszym miejscu jest koncern naftowy Royal Dutch/Shell z 6,8 mld USD zysku (drugi na liście wartości), potem General Motors - odpowiednio 6,7 mld i 36, oraz lider listy Global General Electric z 6,6 mld zysku. Jeśli chodzi o inne firmy informatyczne to wyjątkowo dobrze wypadła tu IBM zajmując 5 miejsce z 6 mld zysku (20 na liście wartości), a tuż za nią na 6 miejscu AT&T - 5,5 mld i 7 miejsce.

Dla uzupełnienia tych różnych rankingów przytoczmy jeszcze kilka danych z listy największych światowych firm produkcyjnych [7]. Obejmuje ona te firmy, które coś wytwarzają, a więc bez usług i pośrednictwa, choć niektóre dane o przychodach (np. dla IBM) są takie same jak na liście Datamation, gdzie obejmowały również usługi. Omawiana lista tworzona jest według całkowitych przychodów firm z wszystkich dziedzin gospodarki, również dla roku 1995. Pierwsze trzy miejsca zajmują na niej korporacje motoryzacyjne: General Motors i Ford Motor z USA (odpowiednio 168,8 i 137,1 mld USD przychodu) oraz japońska Toyota (94,2 mld). Natomiast następne trzy - firmy znane nam z listy Datamation: japońskie Hitachi i Matsushita, produkujące sprzęt elektroniczny i elektryczny (88,6 i 80,6 mld USD przychodu, wartości nieco wyższe niż całkowite przychody na liście Datamation), oraz IBM, pierwsza na tej liście firma w pełni informatyczna o zgodnym z Datamation przychodem. W pierwszej dziesiątce jest jeszcze na siódmym miejscu General Electric (70 mld), a drugą dziesiątkę otwiera niemiecki Siemens (organizacja szersza niż ta z Datamation, 65,3 mld),

aa 15, a na 15, 18 i 20 miejscu są japońskie: Toshiba, Sony i NEC (odpowiednio 57,4, 46,2 i 44,4 mld). Sony nie było na liście Datamation podobnie jak zajmującej 25 miejsce holenderskiej firmy Philips (40,4 mld), bezpośrednio po której idą znane nam firmy japońskie Mitsubishi i Fujitsu (38,4 i 37,8 mld). W pierwszej pięćdziesiątce są jeszcze: Hewlett-Packard i Motorola z USA (odpowiednio miejsce 37 i 31,5 mld oraz miejsce 43 i 27 mld), a także Alcatel z Francji (miejsce 38 i 31,4 mld). Łącznie w tej pięćdziesiątce jest więc 14 firm związanych z informatyką, których przychody stanowiły 29% przychodów pięćdziesiątki.

Wybierając z tej listy firmy o najwyższej stopie zysku stwierdzimy, że będą to firmy z dalekich pozycji listy i różnych dziedzin. Na czele jest tu wydawnictwo brytyjskie News International (91,2% i 923 pozycja) i niemiecka firma tekstylna Vorwerk (44,9% i 930). Z firm elektronicznych w pierwszej dziesiątce znalazły się: na 6 miejscu Telekom Malaysia (31,1% i 784), oraz na 9 miejscu znana z listy Global amerykańska Micron Technology (28,6% i 504).

Podobnie wygląda sytuacja przy kryterium maksymalnego wzrostu zysku. Dwa pierwsze miejsca zajmują tu firmy papiernicze: włoska Cartiere Burgo (97-krotny wzrost i 875 pozycja) oraz japońska Nippon Paper Industries (73-krotny wzrost i 128 pozycja). Na piątym miejscu mamy firmę Quantum z USA (30-krotny wzrost), która była 27 na liście Datamation, a tu dopiero 435, natomiast na miejscu siódmym elektroniczną Mark IV Industries również z USA (wzrost 19-krotny i miejsce 810).

Wszystkie te uszeregowania pokazują, że firmy informatyczne stanowią istotną, aczkolwiek nie dominującą dziedzinę gospodarki światowej.

Różne działy informatyki

Jak już wspomniano największy udział w światowym rynku informatycznym w 1995 roku miały usługi - 27%, tyle co urządzenia peryferyjne w 1985 roku. Szeroko rozumiany dział usług informatycznych (integracja systemów, konsultacje, naprawa uszkodzeń, konserwacja itp.) osiągnął w 1995 roku 184,8 mld USD, o 7,7% więcej niż w roku poprzednim. Tabela 2 przedstawia 20 firm, które osiągnęły największe przychody w dziedzinie usług, podając jednocześnie, jaki procent stanowiły te przychody łącznych przychodów z informatyki.

Tabela 2

Poz. usług	Firma	Poz. l. 100	Kraj	Przychody z usług mln USD	Procent przych. z usług
1	IBM	1	USA	20143,2	28
2	EDS	8	USA	12422,1	100
3	Digital Equipment	7	USA	6497,8	45
4	Hewlett-Packard	3	USA	6257,5	24
5	Computer Sciences	29	USA	3895	95
6	Andersen Consulting	26	USA	3798	90
7	Fujitsu	2	Japonia	3751,9	14
8	Cap Gemini Sogeti	30	Francja	3614,2	95,5
9	Unisys	17	USA	3535,3	57
10	ADP	34	USA	3156,7	100

11	AT&T	9	USA	3016,8	26,5
12	Hitachi	5	Japonia	2917,5	18
13	Siemens Nixdorf	12	Niemcy	2756,9	30,8
14	NEC	4	Japonia	2515,5	13
15	KPMG Peat Marwick	42	USA	2300,3	100
16	NTT Data	23	Japonia	2114,7	40
17	Xerox	24	USA	1815,7	38,9
18	Entex Information Services	52	USA	1785	100
19	Olivetti	18	Włochy	1575,1	26,1
20	Groupe Bull	21	Francja	1484	28

Tak więc, mimo iż procentowy wzrost przychodów osiągniętych z usług nie był w 1995 roku zbyt wysoki, stanowią one dziś poważny odsetek rynku informatycznego, a w Tabeli 2 widzimy zarówno czołowe firmy informatyczne jak IBM, która z usług czerpie 28% przychodów, lub Hewllet-Packard (24%), czy Fujitsu (tylko 14%), a także te, które wyłącznie lub w znacznej części zajmują się tymi usługami jak EDS (100%), CSC (95%) lub Cap Gemini Sogeti (95,5%). Awansowały w stosunku do listy 100 firmy, które mają około połowy przychodów z usług jak Digital Equipment (45%), Unisys (57%) czy NTT Data (40%).

Tabela 3 przedstawia 20 firm osiągających największe przychody ze sprzedaży komputerów osobistych (ko) i stacji roboczych (sr).

Tabela 3

Poz. ko i sr	Firma	Poz. l. 100	Kraj	Przychody z ko i sr mln USD	Procent przych. z ko i sr
1	IBM	1	USA	12949,2	18
2	Compaq Computer	6	USA	9176	62
3	Apple Computer	11	USA	8533,5	75
4	Fujitsu	2	Japonia	6431,5	24
5	Toshiba	10	Japonia	5690,1	50
6	Hewlett-Packard	3	USA	5475,3	21
7	NEC	4	Japonia	5224,5	27
8	Dell Computer	22	USA	4558	86
9	Packard Bell	25	USA	4300	100
10	Sun Microsystem	16	USA	3965	61
11	Gateway 2000	31	USA	3676	100
12	Acer	19	USA	2736	48
13	Digital Equipment	7	USA	2599,1	18
14	AST Research	40	USA	2348,5	100
15	Matsushita	15	Japonia	1826,8	26
16	Olivetti	18	Włochy	1671,7	27,7
17	Hitachi	5	Japonia	1620,8	10
18	Silicon Graphics	39	USA	1397,4	55
19	Siemens Nixdorf	12	Niemcy	1396,4	15,6
20	Groupe Bull	21	Francja	1272	24

Ten dział rynku informatycznego charakteryzował się w 1995 roku znacznym wzrostem sprzedaży komputerów, najczęściej budowanych na procesorach Pentium Intela, która to firma wprowadzała na rynek co kwartał nowy procesor o wyższej częstotliwości zegara. Aczkolwiek przychody z tej dziedziny wzrosły w 1995 roku tylko o 2% osiągając 9,18 mld USD na skutek spadku cen, to firmy, które potrafiły dostosować się do wymagań klientów, zwiększyły swój udział w tym sektorze informatyki i znalazły się w czołówce Tabeli 3. Dotyczy to zajmującej drugie miejsce firmy Compaq, której udział w rynku komputerów osobistych USA wzrósł z 10,6% w 1994 roku do 12,9% w 1995 roku. Podobnie ma się z firmą Packard Bell (analogiczny wskaźnik 10,8% w 1995 roku), z tym że tutaj zadecydował zakup przez Packarda firmy Zenith Data. Również Hewlett-Packard zwiększył swój udział w rynku komputerów osobistych USA ponad dwukrotnie z 3,4 do 7% i jest czwartym dostawcą na tym rynku. Walka cenowa między H-P i Compaqiem kontynuowana była w 1996 roku, kiedy to w marcu Compaq obniżył ceny o 20%, co spowodowało wzrost przychodów z PC i stacji roboczych w I kwartale o 42%, mimo że zysk wzrósł tylko o 8%.

Coraz więcej zadań realizowanych uprzednio przez stacje robocze na procesorach typu RISC spełnianych jest teraz na komputerach osobistych opartych o Pentium, pracujących bądź w systemie Windows NT bądź pod Unixem. Ocenia się, że w 1995 roku w USA 62% komputerów osobistych i stacji roboczych opartych było na procesorze Pentium, a w roku 1996 ten udział wzrośnie do 81%. Projekty następnych procesorów P7 i P7A Intel ma zamiar realizować we współpracy z Hewlett-Packardem.

Mimo iż firma Apple Computer zajmuje trzecie miejsce w Tabeli 3 i utrzymała 11 miejsce na liście Datamation 100, jej udział w tym dziale rynku informatycznego spadł z 9,4% w roku 1994 do 5,4% w roku 1995, a w pierwszym kwartale 1996 poniosła straty w wysokości 749 mln USD i planowała zwolnić 2800 pracowników.

Jeśli chodzi o notebooki to w 1995 roku oczekiwano ilościowego wzrostu o 20%, a rzeczywisty wzrost wyniósł tylko 15%. W roku 1996 Pentium staje się standardem dla komputerów przenośnych i oczekuje się, że w USA 73% tych komputerów będzie miało tej klasy procesory. Małe komputery kieszonkowe nie znalazły jeszcze masowych nabywców. W I kwartale 1996 roku karierę zrobił Palm Pilot firmy US Robotics, która, jak pamiętamy, miała największe tempo przychodów w 1995 roku. Jeszcze mniej można powiedzieć o „sieciowych komputerach osobistych" (Net PC), które mają być sieciowymi terminalami dołączanymi do Internetu. Zapowiedziana jest demonstracja firmy Oracle, a Apple, Compaq IBM i Sun przygotowują swoje projekty. Microsoft zaproponował tu „prosty interakcyjny komputer osobisty" (SIPC), który oprócz dołączenia do sieci steruje wyposażeniem domowym. Niektóre z tych rozwiązań będą mieć cenę mniejszą od 500 USD. Trudno jednak stawiać tu jakiekolwiek prognozy.

Tabela 4 podaje listę 20 firm, które uzyskały największe przychody ze sprzedaży urządzeń peryferyjnych w 1995 roku.

Tabela 4

Poz. u. p.	Firma	Poz. l. 100	Kraj	Prz. z urz. per. mln USD	Procent przych. z urz. peryfer.
1	IBM	1	USA	10071,6	14
2	Hewlett-Packard	3	USA	8604,1	33

3	Seagate Technology	13	USA	7790	95
4	Canon	20	Japonia	5615,7	100
5	Hitachi	5	Japonia	4862,4	30
6	Quantum	27	USA	4174	100
7	Fujitsu	2	Japonia	3215,8	12
8	Matsushita	15	Japonia	3021,3	43
9	Xerox	24	USA	2753,9	59
10	Toshiba	10	Japonia	2731,3	24
11	NEC	4	Japonia	2515,5	13
12	Western Digital	41	USA	2430	100
13	EMC	50	USA	1921,3	100
14	Oki	35	Japonia	1812	59
15	Digital Equipment	7	USA	1732,7	12
16	Storage Technology	49	USA	1639,7	85
17	Acer	19	USA	1596	28
18	Siemens Nixdorf	12	Niemcy	1575,4	17,6
19	Lexmark	55	USA	1478,1	100
20	Mitsubishi	28	Japonia	1367,9	33

Tutaj przede wszystkim zwraca uwagę wysoki wzrost sprzedaży pamięci dyskowych, których w 1995 roku zainstalowano około 1600 terrabajtów za ponad 4 mld USD. Oznacza to 77% przyrost w stosunku do roku 1994. Tak szybki rozwój wraz z postępem technicznym spowodował spadek cen o 37% do 2,72 USD za 1 MB, a w roku 1996 spadek ten ma być jeszcze większy. Zjawisko to eliminuje z rynku słabsze firmy. Seagate Technology nabyła w lutym 1996 roku niedawnego konkurenta Conner Peripherals za 2,9 mld USD, co było jej największym zakupem w ciągu ostatnich trzech lat. Liderem rynku pamięciowego z 41% udziałem jest znajdująca się pośrodku listy 100 i 13 w Tabeli 4 firma EMC, która wyprzedziła IBM (35% udział). Dalej idą: Hitachi (11%), Storage Technology (10%) i Amdahl (3%). Znacznemu wzrostowi tradycyjnych pamięci dyskowych towarzyszyło pojawienie się nowych rozwiązań jak 100 MB napędy Zip firmy Iomega i zastępowanie zwykłych napędów dyskietek napędami optycznymi o pojemności 120 MB (Compaq), czy opracowanie Matsushity i 3M będące stacją 5-krotnie szybszą i o 80-krotnie większej pojemności. Rozwój rynku drukarek był nierównomierny, zwłaszcza tańszych drukarek kolorowych sprzedano mniej niż oczekiwano.

Tabela 5 przedstawia listę 25 firm, które uzyskały największe przychody ze sprzedaży oprogramowania.

Tabela 5

Poz. opr.	Firma	Poz. l. 100	Kraj	Przych. z oprogr. mln USD	Procent prz. z oprogram.
1	IBM	1	USA	12949,2	18
2	Microsoft	14	USA	7418	100
3	Fujitsu	2	Japonia	6431,5	24
4	Computer Associates	33	USA	2460,9	77

5	NEC	4	Japonia	2322	12
6	Novell	45	USA	1897,2	93
7	Oracle	37	USA	1526,9	56,4
8	SAP	51	USA	1349,2	71,5
9	Digital Equipment	7	USA	1299,6	9
10	Hitachi	5	Japonia	1296,7	8
11	Olivetti	18	Włochy	1279,4	21,2
12	Lockheed Martin	44	USA	1234,1	60
13	Siemens Nixdorf	12	Niemcy	1226,3	13,7
14	SAIC	56	USA	910,8	65
15	Hewlett-Packard	3	USA	782,2	3
16	Adobe Systems	85	USA	762,3	100
17	Unisys	17	USA	744,3	12
18	Informix Software	87	USA	714	100
19	Intel	32	USA	648,1	20
20	Sybase	78	USA	640,9	67
21	AT&T	9	USA	569,2	5
22	Autodesk	-	USA	546,9	100
23	SAS Institute	-	USA	534,3	95
24	Groupe Bull	21	Francja	530	10
25	Apple Computer	11	USA	455,1	4

Tu również na pierwszym miejscu pozostaje IBM, mimo że z oprogramowania czerpie ona tylko 18% przychodów, ale już na drugim miejscu jest sprzedająca tylko oprogramowanie firma Microsoft, podobnie jak zajmujące wysokie pozycje Computer Associates (77%), Novell (93%) czy Oracle (56,4%). Ciekawostką jest, że główny dostawca układów scalonych Intel czerpie 20% przychodów informatycznych z oprogramowania, a na niektórych końcowych pozycjach Tabeli 5 znajdują się firmy, które nie występują na liście 100. Zwraca też uwagę niski procent przychodów z oprogramowania takich firm jak Hewlett-Packard (3%) i Apple (4%).

Jako najbardziej perspektywiczny kierunek oprogramowania w najbliższej przyszłości uważa się rozwój komunikacji wewnątrz przedsiębiorstw za pomocą intranetów, czyli sieci korporacyjnych opartych na mechanizmach Internetu. Stosując typową przeglądarkę pracownicy będą mieli dostęp do informacji o przedsiębiorstwie i wspólnie tworzyć zastosowania i publikacje. Będą oni mogli pracować w domu, w podróży. Internety można traktować jako ulepszenie i rozszerzenie zastosowań klient-serwer. Wymaga to jednak przystosowania istniejących sieci, w większości opartych o NetWare, do nowych zadań.

Tabela 6 przedstawia taką samą listę firm, które czerpią największe przychody ze sprzedaży serwerów.

Tabela 6

Poz. serw.	Firma	Poz. l. 100	Kraj	Przych. z serwerów mln USD	Procent prz. z serwerów
1	IBM	1	USA	6474,6	9
2	Hewlett-Packard	3	USA	3650,2	14
3	AT&T	9	USA	3529,1	31
4	Compaq Computer	6	USA	3256	22
5	NEC	4	Japonia	2515,5	13
6	Tandem	43	USA	1846,2	80,8
7	Toshiba	10	Japonia	1820,8	16
8	Digital Equipment	7	USA	1689,4	11,7
9	Fujitsu	2	Japonia	1607,9	6
10	Siemens Nixdorf	12	Niemcy	1226,3	13,7
11	Mitsubishi	28	Japonia	870,5	21
12	Motorola	36	USA	773,3	26
13	Sun Microsystems	16	USA	650	10
14	Silicon Graphics	39	USA	609,8	24
15	Apple Computer	11	USA	568,9	5
16	Hitachi	5	Japonia	486,2	3
17	Data General	62	USA	457,9	38
18	Groupe Bull	21	Francja	424	8
19	Wang Laboratories	69	USA	304,4	27,8
20	Olivetti	18	Włochy	295,7	4,9

Z Tabeli widzimy, że tylko firma Tandem ma ponad 80% przychodów z serwerów. Pozostałe nie osiągają nawet 50%. Wśród tańszych serwerów popularność zdobywa sobie ostatnio system operacyjny Windows NT firmy Microsoft i przewiduje się, że sprzedaż serwerów z tym systemem wzrośnie o 60% w 1996 roku, natomiast serwerów z systemem UNIX tylko o 20%. Systemy z serwerami zastępują duże systemy (mainrames), zwłaszcza przy pakietowym oprogramowaniu handlowym jak R/3 firmy SAP lub osobowo-księgowym firmy PeopleSoft. Również rozwój Internetu przyspiesza rozwój serwerów; przychody z tej dziedziny firmy Sun Microsystems wzrosły w 1995 roku o 21,5% dzięki serwerom Netra Internet. Aby osiągnąć lepszą wydajność opracowuje się systemy 64-bitowe, w czym prowadzi Digital ze swoim AlphaSerwerem.

Tabela 7 przedstawia listę firm o najwyższych przychodach w dziedzinie sprzętu komunikacji danych.

Tabela 7

Poz. k.d.	Firma	Poz. l. 100	Kraj	Przych. ze sprz. k.d. mln USD	Procent prz. z k.d.
1	AT&T	9	USA	3244,5	28,5
2	NTT Data	23	Japonia	3172	60
3	IBM	1	USA	2877,6	4
4	Cisco Systems	38	USA	2667,8	100

5	3 Com	48	USA	1962,8	100
6	Matsushita	15	Japonia	1756,6	25
7	Bay Networks	53	USA	1700	100
8	Motorola	36	USA	1457,3	49
9	Hewlett-Packard	3	USA	1303,7	5
10	Alcatel	64	Francja	1156,6	25
11	US Robotics	70	USA	1091,6	100
12	Mitsubishi	28	Japonia	1077,7	26
13	Cabletron Systems	67	USA	990	90
14	LM Ericsson	65	Szwecja	885,5	80
15	Hitachi	5	Japonia	648,3	4
16	Intel	32	USA	648,1	20
17	Digital Equipment	7	USA	577,6	4
18	Ascom	-	Szwajcaria	560	100
19	Newbridge Networks	95	USA	544,9	85
20	Fujitsu	2	Japonia	536	2

Jest to jedyna z omawianych tu dziedzin informatyki, w której nie przewodzi IBM. Firma ta, uzyskując tylko 4% przychodów z tego działu, ustąpiła tu miejsce dwom firmom telekomunikacyjnym: amerykańskiej AT&T i japońskiej NTT Data. Pierwsza z nich ze sprzedaży sprzętu do przesyłania danych czerpie niespełna 30% swych przychodów, podczas gdy druga - ponad dwukrotnie więcej. Dalej idą już Cisco Systems i 3 Com, które wszystkie swe przychody czerpią z tej dziedziny i które, jak wspomniano, znacznie awansowały na liście 100 w 1995 roku. Cisco obejmuje ponad 70% rynku routerów (układów do marszrutowania sieci komputerowych) i rozszerza swą ofertę na sieci rozległe (WAN) nabywając firmy z tej branży, jak StrataCom. Jest ona czwarta na liście i przychodami ze sprzętu komunikacji danych (2,7 mld USD) dogania IBM.

W roku 1995 protokół przesyłania danych TCP/IP stosowało 46% sieci lokalnych, a stosowany w NetWare Novella IPX tylko 43%. Natomiast sieci typu Token Ring stają się coraz mniej popularne i według przewidywań ich udział spadnie z 25% w 1994 roku do 5,8% w roku 1999. Różnice pomiędzy sieciami lokalnymi i rozległymi zacierają się.

Tabela 8 podaje listę kilkunastu firm, które osiągnęły najwyższe przychody ze sprzedaży wielkich systemów komputerowych. Jest to dziedzina, która nie ma przed sobą wielkich perspektyw, ale na niektórych odcinkach odnosi pewne sukcesy.

Tabela 8

Poz. w. s.	Firma	Poz. l. 100	Kraj	Przych. z wlk. syst. mln USD	Procent prz. z wielk. syst.
1	IBM	1	USA	6474,6	9
2	Fujitsu	2	Japonia	4823,6	19
3	Hitachi	5	Japonia	4376,2	27
4	NEC	4	Japonia	3870	20
5	Unisys	17	USA	1116,4	18

6	Groupe Bull	21	Francja	795	15
7	Siemens Nixdorf	12	Niemcy	769,8	8,6
8	Amdahl	54	USA	758	50
9	Cray Research	89	USA	412,4	61
10	Mitsubishi	28	Japonia	290,2	7
11	Silicon Graphics	39	USA	254,1	10
12	Sequent	-	USA	237,7	44
13	Comparex	90	Niemcy	214,4	32
14	Olivetti	18	Włochy	126,7	2,1

Wszystkie firmy, które miały znaczny udział wielkich systemów w swych przychodach z informatyki z Cray Research na czele nie odnotowały sukcesów na liście 100. Firmy Sequent na tej liście nie było. Tylko Silicon Graphics poprawiła swą pozycję o 8 miejsc, a pierwsza czwórka z Tabeli 8 oraz Siemens i Olivetti utrzymały swe pozycje.

Okazuje się, że rozwój zastosowań typu klient-serwer w pewnych sytuacjach zwiększa zapotrzebowanie na wielkie systemy, takie jak IBM 390 MIPS, których w roku 1995 sprzedano o 60% więcej niż w roku 1994, kiedy to wzrost wyniósł 41% w stosunku do roku 1993. Systemami tymi, które wykorzystują technologię CMOS i są chłodzone powietrzem, zastępuje się stare systemy realizowane w technice ECL i chłodzone wodą, które zużywają ponad 30-krotnie więcej energii i zajmują 10 razy więcej miejsca. Podobnie firma Hitachi opracowała nową rodzinę dużych systemów Skyline, która wykorzystuje hybrydową technologię ECL/CMOS i oferowana jest w cenie średnio 8 mln USD za system. Ceny dużych systemów spadają tak, że ów 60% wzrost ilościowy dał IBM tylko 8,7% wzrostu przychodów z tych systemów. Ocenia się, że średni spadek cen tych systemów wynosi 30% rocznie, to znaczy dwukrotnie więcej niż to miało miejsce w latach osiemdziesiątych. Obniża się też udział wielkich systemów w przychodach z informatyki, który dla IBM wynosił przed pięciu laty 27%.

Uzupełnieniem podanych powyżej danych niech będzie Tabela 9, która przytacza za [3] obroty największych producentów układów scalonych w 1995 roku i ich procentowy wzrost w stosunku do roku 1994. Informatyka jest jednym z głównych odbiorców układów scalonych i rozwój tych obu rynków jest ściśle z sobą związany. Koszt inwestycji w przemyśle półprzewodnikowym jest bardzo wysoki (ok. 1 mld USD za wytwórnię), toteż do końca 1995 roku odczuwano raczej niedobór elementów, co pozwalało utrzymywać ceny zapewniające opłacalność produkcji, a tempo wzrostu określone było zapotrzebowaniem rynku informatycznego.

Tabela 9

Poz.	Firma	Kraj	Procent rynku świat.	Obroty mln USD	Procent wzr. 1994-95
1	Intel	USA	8,7	13172	30
2	NEC	Japonia	7,5	11314	42
3	Toshiba	Japonia	6,7	10077	33
4	Hitachi	Japonia	6	9137	38
5	Motorola	USA	5,8	8732	21

6	Samsung	Korea Pd.	5,5	8329	72
7	Texas Instruments	USA	5,2	7831	41
8	Fujitsu	Japonia	3,7	5538	43
9	Mitsubishi	Japonia	3,5	5272	40
10	Hyundai	Korea Pd.	2,7	4132	172
	Pozostali		44,7	67770	32

Na początku roku 1995 przewidywano spadek produkcji półprzewodników [8], lecz jak wynika z Tabeli 9 nastąpił znaczny wzrost i początkowe przewidywania na rok 1996 mówiły również o 22% wzroście [9]. W pierwszej połowie 1996 roku spadły znacznie ceny układów scalonych zwłaszcza pamięciowych tak, że dla niektórych wytwórców ich produkcja stała się nieopłacalna. Część inwestycji została wstrzymana, a produkcja niektórych układów pamięciowych zamrożona lub nawet zmniejszona. W rezultacie w maju 1996 roku prognozowano korektę wzrostu produkcji półprzewodników w 1996 roku do zaledwie 7,6%, chociaż do końca dekady przewidywano wzrost o 15,4% rocznie [9].

Firmy komputerowe w Polsce

Dla porównania z rankingami światowymi, Tabela 10 podaje za [2] 30 firm o największych przychodach (są one identyczne lub bardzo bliskie przychodom z informatyki). Aby móc porównać przychody dla firm światowych i działających w Polsce podano je również w dolarach, natomiast zysk (netto) podawany jest w złotówkach.

Tabela 10

Poz.	Poz.	Profil	Firma, miasto	Przychody (mln USD)		Zmiana	Zysk	Zyskown.
1995	1994	dział.		1994	1995	%	mln zł	%
1	1	PDEI	Optimus, Nowy Sącz	115,36	172,94	50	13,904	3,3
2	2	DP	JTT Computer, Wrocław	89,73	136,37	52	7,502	2,3
3	3	R	H-P, W-wa/Genewa	68	112,01	65	br. d.	br. d.
4	4	R	IBM, W-wa/Wiedeń	55	82,6	50	br. d.	br. d.
5	5	D	Computer 2000, W-wa	38,95	57,44	47	br. d.	br. d.
6	8	R	ICL Poland, W-wa/Londyn	27,28	56,8	108	br. d.	br. d.
7	6	IOD	Prokom, Gdynia	26,62	49,37	85	22,345	8,7
8	7	IE	ComputerLand Poland, W-wa	26,18	42,12	61	6,445	6,3
9	10	R	Compaq Computer Warszawa/Monachium	23,19	35,64	54	br. d.	br. d.
10	11	D	ABC Data, Warszawa	21,41	31,14	45	br. d.	br. d.
11	33	D	MSP TH'system, Warszawa	16,95	29,65	75	br. d.	br. d.
12	29	ED	Escom Computer Poland, Swarzędz-Jasin	9,68	27,74	187	br. d.	br. d.

13	9	DP	Soft-tronik, Wroclaw/Berlin	24,15	27,26	13	br. d.	br. d.
14	41	ED	Vobis Microcomputer, Szczecin	11,31	27,08	139	br. d.	br. d.
15	13	R	Intel Poland, W-wa/Monachium	19	26,72	41	br. d.	br. d.
16	34	R	Microsoft, W-wa/Monachium	8,6	26	202	br. d.	br. d.
17	15	R	Digital, W-wa/Monachium	17	25,45	50	br. d.	br. d.
18	17	PDE	Baza, Warszawa	16,63	23,01	38	1,523	2,7
19	21	D	System 3000, Kraków	13,24	21,68	64	br. d.	br. d.
20	19	D	California Computer, W-wa	15,14	21,24	40	br. d.	br. d.
21	20	DE	Techmex, Bielsko-Biala	14,26	20,95	47	0,155	0,3
22	18	D	Incom, Wroclaw/Berlin	16,1	20,13	25	br. d.	br. d.
23	109	I	Apexim, Warszawa	8,45	18,99	125	0,756	1,6
24	22	PID	Inwar, Sieradz	12,63	17,1	35	0,441	1,1
25	39	INE	2SI, Katowice	6,27	16,78	168	br. d.	br. d.
26	14	PE	NTT System, Warszawa	17,84	16,77	-6	br. d.	br. d.
27	31	NIO	COIG, Katowice	11,57	16,64	44	3,483	8,6
28	12	PED	Hector, Warszawa	20,72	15,88	-23	br. d.	br. d.
29	42	D	Ab, Wroclaw	7,39	15,58	111	br. d.	br. d.
30	24	R	Novell, W-wa/Dusseldorf	11	15,5	41	br. d.	br. d.

W kolumnie Profil działalności stosowane są następujące oznaczenia: P - produkcja, D - dystrybucja (sprzedaż do pośredników), E - działalność dealerska (sprzedaż do użytkownika końcowego), I - integracja (instalacja i uruchomienie), R - reprezentacja producenta zagranicznego i O - oprogramowanie. Mamy też tu rubrykę Zyskowność, zawierającą wyrażony w procentach iloraz zysku netto do przychodu. Różne są więc dziedziny, na które dzieli się rynek informatyczny na świecie i w Polsce. Wynika to zarówno z różnicy poziomu rozwoju informatyki jak i po prostu innych kryteriów oceny w obu przypadkach. Sto firm informatycznych o największych przychodach w Polsce miało w 1995 roku łączny przychód ok. 1,7 mld USD, co stanowi 0,39% przychodu firm Datamation 100. Jednocześnie przychód „polskiego IBM", czyli firmy Optimus z Nowego Sącza, wynosił w tym samym roku 0,24% przychodu prawdziwej IBM. Tempo rozwoju polskiej informatyki jest szybsze, gdyż przyrost przychodów w latach 1994-95 stu największych firm wyniósł 50,8%, czyli 2,7 razy więcej niż na świecie, doganiamy rynek światowy. Polska lista jest mniej stabilna, choć również 5 pierwszych firm nie zmieniło swej pozycji. W pierwszej setce były jeszcze tylko 2 takie firmy, ale nowych firm pojawiło się 12, wobec 15 na liście Datamation. Zmianę o 1-2 pozycji zanotowano dla 17 firm (na świecie 25), mniejsza natomiast jest tu przewaga spadków nad awansami: 43 do 38, wobec 48 do 25 w Datamation. Wśród największych awansów należałoby wymienić firmę Apexim z Warszawy, która z 109 miejsca w 1994 roku osiągnęła 23 w 1995 i ponad dwukrotnie zwiększyła przychody. Znacznie również

awansowała grupa firm Softbank - z 90 na 32 pozycję, a wzrost przychodów był tu ponad 4-krotny. Wyraźne spadki odnotowały natomiast: Bull Polska - z 30 na 76 miejsce i zmniejszenie przychodów o 47%, CCS - z 26 na 68 i 43%, a z firm poza pierwszą setką fundacja OFEK - z 123 na 238 i 25%.

Jeśli chodzi o polski rynek informatyczny według produktów sprzedaży, a więc nie uwzględniający usług, to dominują w nim komputery osobiste, których udział jednak zmalał z 53,6% w 1994 roku do 48,6% w 1995, następnie idzie oprogramowanie - 18,7% w roku 1994 i 21,2% w roku 1995, drukarki - odpowiednio 16 i 17,1%, serwery - 10,4 i 9,9% i superserwery - 1,3 i 3,2%. Cały ten rynek wzrósł z 750 mln USD w 1994 roku do 1008 mln w 1995 to jest o 34,5%. Natomiast w całym rynku największą dynamikę zanotowano w działalności integratorskiej - 127%, przy czym udział tego działu w rynku komputerowym wzrósł z 12,6% w 1994 roku do 18,3% w 1995. Potem należałoby wymienić dystrybutorów z 84% wzrostem i około 40% udziałem rynku, szkolenia - 71% wzrostu i niespełna 2% udziału i na końcu dealerów oraz producentów sprzętu i oprogramowania - 41-43% wzrostu i odpowiednio 22, 12 i 6% udziału.

Ponieważ największe polskie firmy informatyczne zajmują się głównie produkcją sprzętu, Tabela 11 podaje 15 firm o największych przychodach z tej dziedziny.

Tabela 11

Poz. 1995	Poz. 1994	Firma, miasto	Przych. z pr. sprz. mln zł	Zysk netto mln zł	Zyskown. %
1	1	Optimus, Nowy Sącz	160,63	12,17	3,6
2	2	JTT Computer, Wrocław	42,18	5,82	3,3
3	5	Baza, Warszawa	22,31	1,52	2,7
4	7	BPS, Warszawa	18,97	2,01	6,4
5	4	NTT System, Warszawa	15,04	br. d.	br. d.
6	3	Hector, Warszawa	13,48	br. d.	br. d.
7	6	Inwar, Sieradz	12,89	0,43	1,3
8	-	DTK Computer Polska, Kraków	11,96	0,52	2,8
9	-	Posnet, Warszawa	9,79	br. d.	br. d.
10	8	Gulipin Computers, Wrocław	9,35	br. d.	br. d.
11	10	Format, Warszawa	8,7	br. d.	br. d.
12	-	Telefon 2000, Warszawa	4,38	5,59	19,3
13	-	Medicat-System, Warszawa	4,35	0,03	0,3
14	18	HSK Data, Kraków	3,75	1,62	21,5
15	13	Fideltronik, Zembrzyce	3,37	0,37	10,9

Widzimy, że tylko dwie pierwsze firmy nie zmieniły swych pozycji, a aż czterech nie było na tej liście w 1994 roku. Podawany tu zysk odnosi się do pełnej działalności firm. Wyższą zyskowność uzyskały firmy wyspecjalizowane.

Tabela 12 podaje 20 liderów polskich integratorów, a więc jak wspomniano, najszybciej rozwijającej się dziedziny informatyki w Polsce.

Tabela 12

Poz. 1995	Poz. 1994	Firma, miasto	Przych. z integr. mln zł	Zysk netto mln zł	Zyskown. %
1	1	ComputerLand Poland, Warszawa	73,53	6,45	6,3
2	3	Prokom International, Warszawa	48,12	br. d.	br. d.
3	-	Apexim, Warszawa	41,44	0,76	1,6
4	2	Digital Equipment Polska, W-wa	25,95	br. d.	br. d.
5	15	ATM, Warszawa	21,49	1,56	4,7
6	11	Optimus, Nowy Sącz	20,51	12,17	3,6
7	5	Lumena, Warszawa	20,08	0,06	0,2
8	19	Consortia, Warszawa	14,31	1,13	6,1
9	-	Telefon 2000, Warszawa	12,26	5,59	19,3
10	-	Qumak International, Kraków	12,15	br. d.	br. d.
11	-	AT&T GIS, Warszawa	10,87	br. d.	br. d.
12	-	IBM Polska, Warszawa	10,7	br. d.	br. d.
13	7	COIG, Katowice	10,08	3,48	8,6
14	20	Koncept, Kraków	10,06	0,4	2,5
15	4	CCS, Wrocław	9,43	br. d.	br. d.
16	-	Budimex Soft, Warszawa	8,89	br. d.	br. d.
17	-	Roboatic Microage, Wrocław	8,79	0,18	1,5
18	-	Samba, Gdynia	8,45	0,24	1,8
19	-	Silicon Poland, Warszawa	8,11	0,52	2,6
20	25	Inwar, Sieradz	8,06	0,43	1,3

Widzimy, że sytuacja różni się tu od tej, jaką obserwowaliśmy na listach Datamation, gdzie IBM była pierwszą zarówno ze względu na sumę przychodów jak i przychody z niemal wszystkich dziedzin. W Polsce każdy dział ma swoich liderów i w integracji są to Computerland, Prokom oraz wspomniana już firma Apexim, której nie było na liście integratorów w 1994 roku, a w rok później znalazła się na jej trzecim miejscu. Optimus jest tu dopiero szósty. 45% podanych firm to firmy nowe, które dopiero w 1995 roku wykazały działalność integratorską. Wśród pozostałych najbardziej awansowały firmy: ATM z Warszawy o 10 miejsc i Koncept z Krakowa o 6, natomiast najbardziej spadły firmy: CCS z Wrocławia o 11 miejsc i COIG z Katowic o 6 miejsc, a z następnych 20 firm, których tu nie pokazano, Soft-tronik Service z Warszawy aż o 28 miejsc i Ster-Projekt z Warszawy o 24 miejsca. Rynek integracji ulega więc znacznym zmianom.

Dystrybucja stanowi największą część polskiego rynku informatycznego. Tabela 13 pokazuje 15 firm o największych przychodach z tej dziedziny.

Tabela 13

Poz. 1995	Poz. 1994	Firma, miasto	Przych. z dystr. mln zł	Zysk netto mln zł	Zyskown. %
1	1	Computer 2000 Polska, W-wa	139,25	3,7	2,6
2	10	JTT Computer, Wrocław	133,56	5,82	3,3

3	3	Optimus, Nowy Sącz	78,61	12,17	3,6
4	2	ABC Data, Warszawa	75,5	br. d.	br. d.
5	4	MSP TH'system, Warszawa	71,88	br. d.	br. d.
6	8	System 3000, Kraków	52,06	br. d.	br. d.
7	5	California Computer, Warszawa	51,5	br. d.	br. d.
8	7	Techmex, Bielsko-Biała	45,72	0,16	0,3
9	6	Soft-tronik Polska, Wrocław	45,5	br. d.	br. d.
10	-	JTT Computer SEC, Warszawa	41,33	1,33	3,2
11	14	Ab, Wrocław	36,65	br. d.	br. d.
12	13	Incom Team, Wrocław	31,73	br. d.	br. d.
13	-	JTT Computer II, Katowice	27,05	0,23	0,8
14	-	Escom Computer, Swarzęc-Jasin	26,9	br. d.	br. d.
15	9	TCH Components, Warszawa	26	br. d.	br. d.

Mamy tu 20% firm nowych (w drugiej piętnastce jest ich prawie 50%), wśród których dominują terenowe filie JTT Computer, a wrocławska centrala tej firmy awansowała z 10 na 2 miejsce. Spadły firmy: TCH Components o 6 miejsc i Soft-tronik o 3 miejsca. Optimus zajmował zarówno w 1994 jak i 1995 roku 3 miejsce, gdyż jest to firma nastawiona na działalność dealerską co widać z Tabeli 14, pokazującej liderów tej dziedziny w Polsce.

Tabela 14

Poz. 1995	Poz. 1994	Firma, miasto	Przych. z dz. deal. mln zł	Zysk netto mln zł	Zyskown. %
1	1	Optimus, Nowy Sącz	82,02	12,17	3,6
2	-	Vobis Microcomputer, Szczecin	59,1	br. d.	br. d.
3	-	Escom Computer, Swarzęc-Jasin	40,35	br. d.	br. d.
4	5	ComputerLand Poland, Warszawa	27,98	6,45	6,3
5	11	Koma, Katowice	22,39	0,88	3
6	7	NTT System, Warszawa	19,51	br. d.	br. d.
7	3	Hector, Warszawa	17,33	br. d.	br. d.
8	-	Softbank, Warszawa	16,48	4,39	13,6
9	10	Ster-Projekt, Warszawa	16,34	1,32	5,7
10	-	Sprint, Olsztyn	14,89	0,19	1,3
11	13	Silcon Poland, Warszawa	12,16	0,52	2,6
12	-	Karen, Sulejówek	12,15	br. d.	br. d.
13	-	Prokom International, Warszawa	11,85	br. d.	br. d.
14	-	KEN, Wrocław	10,59	br. d.	br. d.
15	-	Unilot, Warszawa	9,81	br. d.	br. d.

Tutaj ponad 50% stanowią firmy nowe i niektóre z nich jak Vobis i Escom uplasowały się zaraz za Optimusem. Koma awansowała o 6 miejsc, a Silcon o dwa miejsca, natomiast firma Hector spadła o 4 miejsca. Również ten dział rynku jest daleki od stabilności.

Szukając firm o największej zyskowności natrafimy na małe firmy jak RoboBAT z Krakowa (32,4%) i Young Digital Poland z Gdańska (28,8%), które nie były klasyfikowane pod tym względem w 1994 roku, czy Logotec Engineering z Mysłowic (27,3%) która awansowała 12 na 13 miejsce.

Natomiast wyszukując firmy o najwyższym procentowym wzroście przychodów znajdziemy na pierwszym miejscu firmę Prosnet z Warszawy - 7,7-krotny wzrost, następnie JTT Computer VI z Krakowa - 3,5-krotny i Softbank z Warszawy - 3,3-krotny. Prognozy na rok 1996 pokazują, że wzrost przychodów 1995/96 będzie dla większości firm kilkakrotnie mniejszy w stosunku do roku poprzedniego, tylko firma Westek Polska z Warszawy ma mieć ten parametr ponad dwukrotnie wyższy.

W ostatniej przytaczanej tu Tabeli 15 podajemy listę firm czerpiących największe przychody z tworzenia oprogramowania w Polsce.

Tabela 15

Poz. 1995	Poz. 1994	Firma, miasto	Przych. z tw. opr. mln zł	Zysk netto mln zł	Zyskown. %
1	1	Prokom Software, Gdynia	35,91	br. d.	br. d.
2	3	Softbank, Warszawa	15,51	4,39	13,6
3	2	CSBI, Warszawa	9,34	1,6	6
4	-	CrossComm Poland, Gdańsk	7,92	0,29	3,1
5	-	Computron-Rewiks, W-wa	7,86	br. d.	br. d.
6	5	COIG, Katowice	5,24	3,48	8,6
7	-	River, Kraków	5,1	br. d.	br. d.
8	4	Centrum Inform. Energ., W-wa	4,12	br. d.	br. d.
9	16	Compact Disc Novelty, Kraków	3,21	0,44	11,4
10	18	Simple, Warszawa	3,17	0,11	2,4
11	20	Logotec Engineering, Mysłowice	2,99	1,48	27,3
12	14	Unisoft, Gdynia	2,98	br. d.	br. d.
13	9	MacroSoft, Warszawa	2,89	0,53	9,2
14	-	Kom-Pakt, Warszawa	2,8	0,22	4,6
15	-	Apexim, Warszawa	2,76	0,76	1,6
16	-	Telefon 2000, Warszawa	2,64	5,59	19,3
17	-	Young Digital Poland, Gdańsk	2,43	0,7	28,8
18	-	Polnet Technologies, Warszawa	2,41	br. d.	br. d.
19	22	Koncept, Kraków	2,36	0,4	2,5
20	29	Jason MacKenzie, Łódź	2,35	-0,04	-0,4

Tu też obserwujemy duże zmiany - 35% firm nowych i tylko jedna (lider Prokom Software) zachowująca dawną pozycję. Często polskie firmy softwarowe nawiązują kontakty z firmami zachodnimi. Jeśli chodzi o głównych odbiorców krajowego oprogramowania to w 1995 roku 31,8% stanowił przemysł, 18,3% - banki, 12,9% - administracja, 11,6% - użytkownicy indywidualni, a 10,1% - handel. Natomiast oprogramowania zagranicznego sprowadziliśmy w 1995 roku za 54,6 mln USD, co stanowiło 73% przychodów z tworzenia oprogramowania w Polsce w tym samym roku. Oprogramowanie Microsoftu stanowiło 48% sprowadzonego oprogramowania zagra-

nicznego, Novella - 28%, a firmy Computer Associates - 11%. Wzrost wartości sprowadzanego oprogramowania w latach 1994/95 wyniósł 76%, ale dla Microsoftu był ponad 3-krotny. Wśród Unix'owych systemów zarządzania bazami danych (łącznie wartości 22,8 mln USD w 1995 roku i ponad dwukrotny wzrost w stosunku do roku 1994) dominuje oprogramowanie firmy Informix - 46% tych systemów i ponad 5-krotny wzrost w okresie 1994/95, potem Oracle - odpowiednio 35 i 40% i Progress - 11 i 26,7%.

Jeśli chodzi o najpopularniejsze języki programowania aplikacji w Polsce to w 1995 roku około 36% programów pisanych było w językach C lub C++, po około 13% w Pascalu i Clipperze i po około 9% w językach wspomnianych baz danych Informix i Progress. Rok wcześniej na pierwszym miejscu był Clipper - 26%, potem C/C++ - 24%, Pascal -14%, Informix - 6% i Progress - 4%. Natomiast przyglądając się ofercie krajowych aplikacji według systemów operacyjnych w ostatnich latach stwierdzamy, że wciąż dominuje w nich DOS, przy czym jego udział waha się od 43% w 1994 roku do 29% w latach 1993 i 1995, można więc mówić ostatnio o tendencji malejącej. Na drugim miejscu w 1995 roku znalazł się Unix - 25%, przy czym miał on wyższy udział w latach 1992-93. Dalej idzie sieciowy Novellowski DOS/NetWare - 22-24% w latach 1992-93 i 1995, a tylko 13% w roku 1994. Aplikacje Windowsowe wzrosły z nieco ponad 5% w latach 1992-93 do ponad 13% w okresie 1994-95. Z innych systemów można zauważyć wzrost OS/2 - do 4% w 1995 roku i pojawienie się Windows NT - ponad 3% w tymże roku, natomiast zmniejszanie się udziału systemu Mac - do niewiele ponad 1% w 1995 roku.

W dziedzinie komputerów osobistych obserwowano w Polsce w roku 1995 podobne tendencje co na rynku światowym - spadek cen i wzrost sprzedaży. Łącznie sprzedano 327 tysięcy sztuk, co oznacza 20% wzrost w stosunku do roku 1994. Najwięcej dostarczył ich Optimus - 130 tys i 16% wzrostu, następnie JTT - 30 tys i 33% spadek, Compaq -16 tys i wzrost o 26%, oraz Vobis - 15 tys i wzrost o 85%. Największy wzrost, ponad 11-krotny odnotowała firma NTT dostarczając w roku 1995 13,3 tys., prawie potroiła też dostawy firma Dell - 10 tys. Standard sprzedawanych komputerów niewiele odbiega od poziomu światowego - w 1996 roku są to systemy na procesorach Pentium o coraz wyższych częstotliwościach zegara.

Na polskim rynku drukarek zanotowano w 1995 roku dalsze zmniejszanie udziału drukarek mozaikowych, który niewiele przekraczał połowę tego rynku. Jednocześnie nastąpiły pewne zmiany wśród dystrybutorów tych urządzeń. Firmy Epson i Star, dominujące dotychczas w tym zakresie, zaczęły tracić na rzecz Oki i Panasonica. Prognoza na rok 1996 mówi o 27% wzroście wartościowym rynku drukarek w stosunku do 1995 roku. Wzrost ilościowy ma wynieść 20,3% w związku ze wzrostem udziału droższych drukarek atramentowych i laserowych. Te pierwsze mają stanowić ponad 33% wszystkich sprzedanych w 1996 roku drukarek, a drugie - ponad 31%. Drukarki mozaikowe mają spaść na trzecią pozycję z udziałem 28%. Przewiduje się też niewielki wzrost udziału drukarek ciężkich (ponad 7%), które w Polsce są w większości urządzeniami uderzeniowymi. Wśród dystrybutorów drukarek atramentowych i laserowych dominuje firma Hewlett-Packard, przy czym wydaje się, że w przyszłości technika laserowa będzie rozwijać się szybciej.

Ostatnim działem polskiego rynku informatycznego, opisanym dokładnie w pracy [2], są szkolenia, które są u nas jeszcze słabo rozwinięte - zaledwie nieco ponad 1% tego rynku. Dlatego dział ten, jak wspomniano, rozwija się dosyć szybko, rośnie ilość

placówek szkolących i słuchaczy. Znane szkoły jak Multitrade tworzą nowe oddziały terenowe, powstają nowe centra jak Edusoft czy firmy jak Global Knowledge Network. Najwięcej jest autoryzowanych ośrodków szkoleniowych Microsoftu i Novella. Działalność szkoleniową prowadzą też duże firmy międzynarodowe jak Digital, Hewlett-Packard i ICL, a chce prowadzić Compaq; popularne są też kursy prowadzone za granicą. Najwięcej przychodów ze szkoleń (prawie 2,5 mln zł) uzyskała w 1995 roku firma Multitrade z Warszawy, która przeszkoliła 6 tysięcy słuchaczy. Na drugim miejscu znalazł się IKIK również z Warszawy - odpowiednio prawie 2,2 mln zł i 3000 słuchaczy. Obie te instytucje stały tak samo na czele ośrodków szkolących w 1994 roku. Natomiast na 3 miejscu jest firma 2SI z Katowic (poprzednio Techmex), która uzyskała ze szkoleń ponad 2 mln zł mimo, że stanowiło to tylko 5% ogólnych przychodów firmy. Dalej idzie sieć ośrodków EIITE należących do grupy Soft-tronik - 1,9 mln zł i 2700 słuchaczy i ośrodki firm międzynarodowych: ICL Poland - ponad 1,8 mln zł i ponad 1800 słuchaczy, Hewlett-Packard Polska - ponad 1,5 mln i 930 oraz Digital Equipment Polska - prawie 1,3 mln i 800. Przychody ze szkoleń wynosiły w tych ostatnich firmach 4-7% przychodów ogólnych, natomiast Multitrade i EIITE miały 100% przychodów ze szkoleń, a IKIK - 98%.

Podsumowanie

Przytoczone tu liczby odnośnie światowego i polskiego rynku informatycznego pozwalają na sformułowanie pewnych wniosków i porównań. Niektóre z nich podawane już były poprzednio, ale warto je na koniec uwypuklić i rozszerzyć. Światowy rynek komputerowy rozwijał się w ostatnich latach dosyć dynamicznie, choć najnowsze prognozy [10] mówią o przyhamowaniu tego tempa w najbliższej przyszłości. O ile bowiem w roku 1995 przychody tego rynku wzrosły o prawie 20% w stosunku do przychodów z roku poprzedniego, to przewidywania na lata 1996-8, liczone coprawda tylko odnośnie sprzętu, wynoszą tylko12,5%, a na lata następne jeszcze trochę maleją. Natomiast polski rynek informatyczny jest mniej rozwinięty w stosunku do rynku światowego, stąd wyższe tempo rozwoju - około 30% w latach 1994/95 i to licząc w USD, a dla 200 największych firm komputerowych nawet ponad 70%. Również w najbliższych latach spodziewać się można dalszego rozwoju tego rynku w tempie wyższym od światowego. Natomiast trudno jest porównywać strukturę rynku informatycznego w Polsce i na świecie, gdyż w dostępnych danych zastosowano inne kryteria podziału. Widać jednakże, że jako świeżo rozwijający się rynek polski ma on większy udział w sprzęcie - dystrybucja i składanie komputerów, głównie osobistych. Na podstawie [2], uwzględniając cały rynek informatyczny (łącznie z usługami, konsultacjami, szkoleniem), można ocenić, że udział PC wynosił nieco ponad 40% w roku 1994 i niespełna 38% w roku 1995, był więc prawie dwukrotnie wyższy niż na świecie. Udział serwerów był w 1995 roku w Polsce o około 28% wyższy niż na rynku światowym, a oprogramowania - o prawie 50%. Natomiast udział usług był u nas niższy o około 17% i mimo znacznego wzrostu przychodów firm integratorskich w latach 1994/95, nie widać z przytoczonych w [2] ogólnych danych, poprawy sytuacji w tym okresie. Udziału urządzeń peryferyjnych nie można porównać, gdyż dla Polski podane są tylko dane dla drukarek. Tak samo przesyłanie danych oceniane jest poprzez przytoczenie liczb charakteryzujących całą polską telekomunikację. Natomiast dane odnośnie szkolenia są dostępne tylko dla rynku polskiego.

Polski rynek informatyczny „dogania" więc światowy rynek komputerowy zachowując wyższe tempo rozwoju i mając wyższy udział sprzętu. Opóźnienie w usługach jest zmniejszane przez szybki rozwój integracji systemów, natomiast niepokoi niskie tempo rozwoju produkcji oprogramowania.

Literatura

[1] Datamation 100, Datamation 12/96, str. 32
[2] Raport Specjalny Computerword Top 200, Polski Rynek Komputerowy 1995, czerwiec 1996
[3] Z. Blewoński: Krach na rynku Komputerowym?, PC Kurier 16/96, str. 89
[4] E. Dejesus: Irresistible Drives, Byte 6/96, str. 91
[5] Sun kupuje Cray'a, PC Kurier 16/96, str. 145
[6] T. Kulisiewicz: Informatyka na topie, PC Kurier 16/96, str. 92
[7] 1000 World's Largest Manufacturing Companies, Industry Week 10/96, str. 12
[8] J. Ryżko: Elektronika światowa w 1995 roku i latach następnych, Informatyka 8/95 str.1
[9] A. Machol: Koniec błogiego spokoju, PCKurier 11/96 str. 117
[10] R. Weiss: Embedded Systems - Applications Overview, Computer Design 10/96 str. 58