SYMULACJE KOMPUTEROWE W BADANIACH I ROZWOJU

pod redakcją naukową
Zenona A. Sosnowskiego

Politechnika Białostocka
Białystok 2020

SYMULACJE KOMPUTEROWE W BADANIACH I ROZWOJU

pod redakcją naukową

Zenona A. Sosnowskiego

Politechnika Białostocka

OFICYNA WYDAWNICZA POLITECHNIKI BIAŁOSTOCKIEJ

BIAŁYSTOK 2020

Monografia sfinansowana ze środków Polskiego Towarzystwa Symulacji Komputerowej.

Recenzenci:
dr hab. inż. Andrzej Grzyb, em. prof. PK
dr hab. inż. Bolesław Szafrański, prof. WAT

Redaktor naukowy dyscypliny informatyka techniczna i telekomunikacja:
prof. dr hab. Jarosław Stepaniuk

Redakcja i korekta językowa:
Katarzyna Duniewska

Projekt okładki:
Marcin Dominów

Redakcja techniczna, skład, grafika:
Oficyna Wydawnicza Politechniki Białostockiej

ISBN 978-83-66391-27-7
ISBN 978-83-66391-28-4 (eBook)
DOI: 10.24427/978-83-66391-28-4

Druk: PARTNER POLIGRAFIA Andrzej Kardasz

Oficyna Wydawnicza Politechniki Białostockiej
ul. Wiejska 45C, 15-351 Białystok
tel.: 85 746 91 37
e-mail: oficyna.wydawnicza@pb.edu.pl
www.pb.edu.pl

Spis treści

Wstęp 7

Modelowanie i symulacja niegładkiego zagadnienia ruchu
Wiesław Grzesikiewicz, Artur Zbiciak, Katarzyna Rutczyńska-Wdowiak 9

Stochastyczny model ogólnego cyklu życia ataku cybernetycznego
Romuald Hoffmann 41

Układ żył w palcu i odcisk palca w multimodalnym systemie biometrycznym
Maciej Szymkowski, Khalid Saeed 51

Uczenie maszynowe w detekcji zmian patologicznych na obrazach okulistycznych
Maciej Szymkowski, Khalid Saeed, Emil Saeed 65

Wizualizacja algorytmów wyszukiwania wzorców liniowych w obrazach 2*D*
Paweł Zabielski 75

Wybrane strategie walki z przeciwnikiem w turowej grze strategicznej
Adam Dąbrowski, Magdalena Topczewska 83

Kompakcja wejściowego strumienia bitowego z użyciem metod zbiorów przybliżonych do symulacji mocy w układach sekwencyjnych
Tomasz Grześ 99

Problem komunikacji federatów w symulacji opartej o HLA
Krzysztof Panufnik 113

Metoda wyznaczania stanu agentów w symulacji wieloagentowej o zmiennej rozdzielczości
Dariusz Pierzchała, Przemysław Czuba 125

Symulator systemów klasy SOA
Adrian P. Woźniak 141

Optymalizacja procesu magazynowania wysokoskładowego
Piotr Kisielewski, Przemysław Talarek 151

Symulacyjne badanie efektywności funkcjonowania przedsiębiorstwa serwisującego odzież roboczą
Tadeusz Nowicki, Robert Waszkowski 171

Sieci neuronowe w przetwarzaniu obrazów: przegląd wybranych osiągnięć
Karol Przybyszewski 193

Contents

The modeling and simulation of non-smooth motion problem
Wiesław Grzesikiewicz, Artur Zbiciak, Katarzyna Rutczyńska-Wdowiak 39

Stochastic model of the general cyber-attack life cycle
Romuald Hoffmann 50

Finger veins and fingerprint in multimodal biometrics system
Maciej Szymkowski, Khalid Saeed 63

Machine Learning in the detection of pathological changes in ophthalmic images
Maciej Szymkowski, Khalid Saeed, Emil Saeed 73

Visualization of linear patterns search algorithms in 2D images
Paweł Zabielski 82

Selected battles strategies in turn-based strategy game
Adam Dąbrowski, Magdalena Topczewska 97

Bitstream compaction using rough set theory for power simulation in sequential circuits
Tomasz Grześ 111

Federate communication problem in HLA-based simulation
Krzysztof Panufnik 124

The method of state estimation in multiagent multiresolution simulation
Dariusz Pierzchała, Przemysław Czuba 138

Simulator of SOA class systems
Adrian P. Woźniak 149

Optimization of the high-level storage process
Piotr Kisielewski, Przemysław Talarek 170

Simulation method for effectiveness evaluation in a working clothes rental company
Tadeusz Nowicki, Robert Waszkowski 191

Neural Networks in Computer Vision: a review of selected advancements
Karol Przybyszewski 205

Wstęp

Symulacje komputerowe zjawisk występujących w przyrodzie i technice znajdują coraz bardziej powszechne zastosowanie i zdobywają popularność. Techniki symulacyjne są przydatne w efektywnym badaniu złożonych systemów szczególnie tam, gdzie metody analityczne byłyby zbyt pracochłonne, a niekiedy nawet niemożliwe do wykorzystania.

Autorzy rozdziałów niniejszej książki prezentują szeroki zakres tematyczny wykorzystania metod modelowania i symulacji komputerowej. Materiał został podzielony na trzy zasadnicze części obejmujące prace badawcze, prace o charakterze badawczo-rozwojowym oraz prace aplikacyjne.

Początkowe rozdziały prezentują wyniki pięciu prac badawczych. Pierwsza z nich opisuje modelowanie i symulację niegładkiego zagadnienia ruchu układu mechanicznego na przykładzie koła, którego ruch jest ograniczany. W rozdziale następnym zaproponowano stochastyczny model ogólnego cyklu życia ataku cybernetycznego zakładając, że czasy trwania poszczególnych faz ataku mają dowolny ciągły rozkład prawdopodobieństwa. Kolejne dwie prace obejmują zagadnienia biometrii – w szczególności zaproponowany został algorytm do detekcji wysięków twardych na obrazach siatkówki oka z zastosowaniem metod uczenia maszynowego. Problematyce uczenia maszynowego poświęcona została również ostatnia praca tej części. Przedstawiono w niej nowy algorytm wykrywania płaskich wzorców, który ma zastosowanie również w przypadku danych wielowymiarowych.

Kolejne rozdziały to prace o charakterze badawczo-rozwojowym. W pierwszym opracowaniu z tej części (rozdział szósty) przedstawiono porównanie wybranych podejść strategii gry turowej. Kolejny rozdział przedstawia użycie metod zbiorów przybliżonych do symulacji mocy w układach sekwencyjnych. Rozdział ósmy przedstawia problem komunikacji między federatami symulatora opartego o standard HLA (ang. *High Level Architecture*). W kolejnym rozdziale zaproponowano wykorzystanie technik wieloagentowych w symulacji o zmiennej rozdzielczości, przy czym w agregacji oraz deagregacji stanu agenta zastosowano algorytmy konsensusu oraz sterowania formacją. W następnym rozdziale przedstawiono symulator systemów klasy SOA (ang. *Service Oriented Architecture*).

Końcowa część książki zawiera dwie prace o charakterze aplikacyjnym. Pierwsza z nich opisuje sposoby optymalizacji procesu magazynowania wysokiego składowania. W drugiej Autorzy przedstawiają modele procesów biznesowych przedsiębiorstwa.

Ostatnia praca ma charakter przeglądowy. Opisuje wysokopoziomowe koncepcje i architektury sieci neuronowych oraz łączenie tej wiedzy z praktycznymi obszarami zastosowania technik przetwarzania obrazów. Zespół autorski niniejszej monografii ma nadzieję, że treści w niej zawarte zainteresują potencjalnych czytelników.

Wiele osób pomogło w przygotowaniu i publikacji tej książki. Chciałbym podziękować: Autorom poszczególnych rozdziałów, Recenzentom za pomoc w procesie selekcji materiału, a Polskiemu Towarzystwu Symulacji Komputerowej za wspieranie naszych wysiłków.

Zenon A. Sosnowski

Białystok, maj 2020

DOI: 10.24427/978-83-66391-28-4_1

Modelowanie i symulacja niegładkiego zagadnienia ruchu

Wiesław GRZESIKIEWICZ[*]
Artur ZBICIAK[*]
Katarzyna RUTCZYŃSKA-WDOWIAK[**]

1. Wprowadzenie

W klasycznym (newtonowskim) zagadnieniu ruchu układu mechanicznego o N stopniach swobody, wyznacza się – w ustalonym przedziale czasu – rozwiązanie równania różniczkowego:

$$\ddot{X} = f\left(t, X, \dot{X}\right), \quad t \in \left[0, t_{end}\right) \tag{1a}$$

spełniające warunki początkowe:

$$X\left(0\right) = X_{\mathrm{o}}, \quad \dot{X}\left(0\right) = V_{\mathrm{o}} \tag{1b}$$

W tak sformułowanym zagadnieniu początkowym zakłada się, że funkcja f: $R^3 \rightarrow R^N$ jest ciągła i spełnia dla drugiego i trzeciego argumentu warunek Lipschitza. Wtedy istnieje jednoznaczne rozwiązanie opisujące ruch układu, a funkcja X jest ciągła i dwukrotnie różniczkowalna, czyli $X \in C^2([0, t_{end}), R^N)$.

Niegładkość zagadnienia ruchu oznacza, że funkcja f jest nieciągła lub nieróżniczkowalna względem drugiego lub trzeciego argumentu. Tego rodzaju zagadnienie dotyczy np. układu mechanicznego z więzami jednostronnymi lub z siłami tarcia suchego. W monografii [1], w której jest analizowany układ z więzami jednostronnymi pokazano, że rozwiązanie zagadnienia istnieje w klasie funkcji absolutnie ciągłych $X \in C_{ab}([t_{\mathrm{o}}, t_{end}), R^N)$. To oznacza, że pochodna $\dot{X}$ jest nieciągła, a więc należy sformułować dodatkowe zadanie określające nieciągłe zmiany prędkości.

W rozważanym dalej niegładkim zagadnieniu ruchu koła rozpatrujemy niedoskonałe więzy jednostronne. Niedoskonałość ta oznacza, że z siłą reakcji więzów jest stowarzyszona prostopadła do niej siła tarcia suchego. Tego rodzaju hipoteza została postawiona pod koniec XIX w. przez Painlevé'a [2, 3]. Współczesna analiza tego zagadnienia opiera się na podstawach analizy niegładkiej prezentowanych w pracach Jeana,

[*] Politechnika Warszawska
[**] Politechnika Świętokrzyska

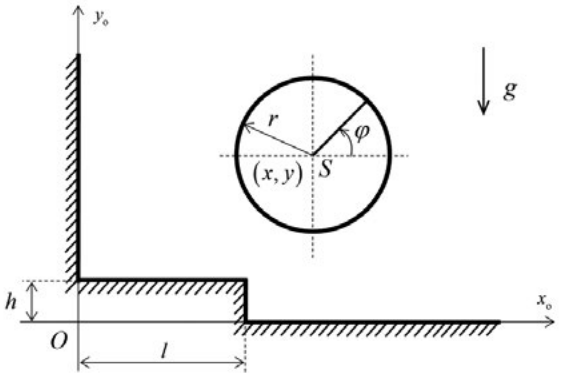
Moreau i Panagiotopoulosa [4-8], a także w pracach autorów niniejszego artykułu, związanych z modelowaniem maszyn i pojazdów, np. [9-11]. Należy również wspomnieć o współczesnych pracach [12, 13, 14], w których rozpatruje się osobliwe zagadnienie nazywane paradoksem Painlevé'a, szeroko dyskutowane na początku XX w. przez Prandtla, Hamela, Kleina i Misesa [2].

W stosunku do wymienionych wyżej prac rozpatrujemy specyficzną metodę opisu ruchu z niedoskonałymi więzami jednostronnymi, przystosowaną do analizy stosunkowo prostego zagadnienia związanego z płaskim ruchem sztywnego koła. Tak wybrane zagadnienie w pełni ilustruje niegładką problematykę więzów jednostronnych, a jednocześnie umożliwia pominięcie sytuacji, w której powstaje paradoks Painlevé'a.

Poza sformułowaniem zagadnienia ruchu prezentujemy również metodę wyznaczania rozwiązania tego zagadnienia, a także przedstawiamy wyniki symulacji ruchu koła.

2. Geometryczne i kinematyczne cechy układu

Rozpatrujemy nieodkształcalne koło, którego ruch w polu grawitacyjnym jest ograniczany przez nieodkształcalne płaszczyzny o szorstkiej powierzchni. Na rysunku 1 pokazano ilustrację rozważanego układu oraz zaznaczono trzy współrzędne opisujące ruch koła, czyli: (x, y) – współrzędne środka koła S oraz φ– kąt obrotu koła.



RYS. 1. Geometryczna ilustracja rozważanego układu koło–więzy
FIG. 1. The geometric illustration of the considered wheel-ties arrangement

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Ograniczenia ruchu na rysunku 1 są przedstawione za pomocą pogrubionej linii, która reprezentuje granicę obszaru dopuszczalnych położeń koła na płaszczyźnie Ox_0y_0. Na rysunku 1 zaznaczono również główne wymiary linii granicznej oraz koła, czyli r, h, l, przy czym rozważamy przypadek, gdy $r > h$ oraz $l > r$. Poza tym zaznaczono wektor pola grawitacji g.

Ze wstępnej analizy rysunku 1 wynika, że rozważane ograniczenia można określić obszarem dopuszczalnych położeń środka koła, gdyż trzecia współrzędna, czyli kąt obrotu koła φ, nie jest ograniczana.

Na podstawie geometrycznej analizy rysunku 1 ustalamy opis wspomnianego ograniczenia w postaci relacji:

$$(x(t), y(t)) \in \Omega \subset R^2 \qquad \forall t \in [0, t_{end}) \tag{2}$$

gdzie zbiór Ω, określający obszar dopuszczalnych położeń środka koła, ma postać:

$$\Omega := \{(x, y): x \geq r; y \geq \Phi(x)\} \tag{3}$$

jeśli funkcja Φ jest określona następująco:

$$\Phi(x) := \begin{cases} h + r & \text{gdy} \quad r \leq x \leq l_1 \\ f(x) & \text{gdy} \quad l_1 < x \leq l_1 + a \\ r & \text{gdy} \quad x > l_1 + a \end{cases} \tag{4a}$$

gdzie:

$$l_1 := l - r\,, \; a = \sqrt{r^2 - (r-h)^2} \tag{4b}$$

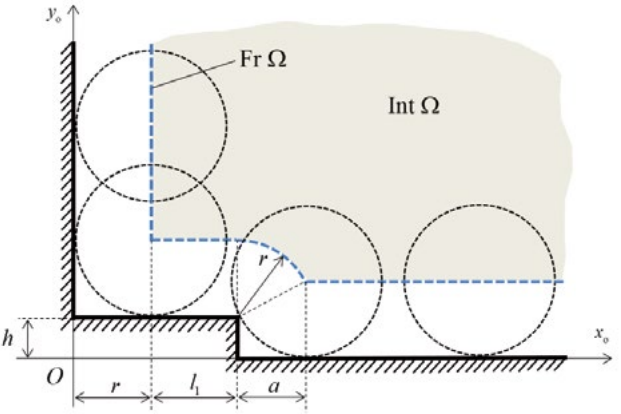
$$f(x) := \sqrt{r^2 - (x - l_1)^2} + h \tag{4c}$$

Na rysunku 2 pokazano zbiór Ω oraz zaznaczono szarym kolorem obszar wnętrza zbioru Ω, czyli Int Ω, a linią przerywaną jego granicę – Fr Ω. Opisy tych zbiorów ustalono na podstawie wzorów (3) i (4), tzn.:

$$\text{Int}\,\Omega := \left\{(x, y): \quad x > r; \; y > \Phi(x)\right\} \tag{5a}$$

$$\text{Fr}\,\Omega := \left\{(x, y): \quad \left\{y \geq h + r, \; x = r\right\} \cup \left\{y = \Phi(x), \; x > r\right\}\right\} \tag{5b}$$

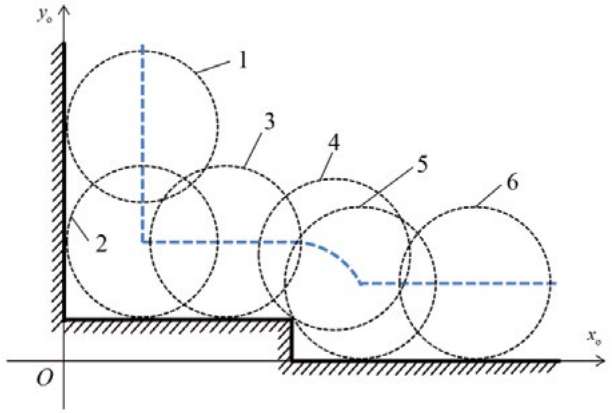
Dodatkowo na rysunku 2 zaznaczono położenia koła w wybranych sektorach granicy Fr Ω. Bardziej szczegółowy wykres położeń koła, we wszystkich charakterystycznych sektorach granicy Fr Ω, pokazano na rysunku 3.



RYS. 2. Wykres zbioru Ω i jego granicy Fr Ω oraz kilka szczególnych położeń, w których koło styka się z brzegiem ograniczenia

FIG. 2. The chart of the set Ω and its border Fr Ω and a few special positions in which the wheel is in contact with the edge of the constraint

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 3. Sześć szczególnych położeń, w których koło styka się z brzegiem ograniczenia

FIG. 3. Six specific positions in which the wheel is in contact with the edge of the constraint

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Aby opis ograniczenia ruchu był pełny, należy obok zbioru Ω ustalić również implikacje kinematyczne, które wynikają z relacji (2) i będą nazywane jej następstwami. W pracy [3] szczegółowo rozpatrywano następstwa, które wyznaczają zbiory dopuszczalnych prędkości i dopuszczalnych przyspieszeń.

Jeżeli koło (por. rys. 2) znajduje się w pozycji takiej, że $(x, y) \in \text{Int}\ \Omega$, czyli gdy koło nie styka się z brzegiem ograniczenia, to wtedy wspomniane wyżej ograniczenia kinematyczne nie występują, czyli koło porusza się swobodnie.

Gdy koło styka się z brzegiem ograniczenia, czyli $(x, y) \in \text{Fr}\ \Omega$, to wtedy prędkość środka koła (v_x, v_y) jest ograniczana, gdyż powinna spełniać relację:

$$\left(v_x, v_y\right) \in \mathsf{D}\Omega\left(x, y\right) \subset R^2, \quad \left(x, y\right) \in \text{Fr}\ \Omega\,, \tag{6}$$

gdzie $\mathsf{D}\Omega$ jest odwzorowaniem nazywanym pierwszym następstwem relacji (2). Definicję odwzorowania $\mathsf{D}\Omega$ zamieszczono w pracy [1], natomiast szczegółowy opis tego odwzorowania w rozpatrywanym zadaniu będzie podany dalej.

Jeżeli położenie i prędkość środka koła spełniają relację (6), to wtedy może nastąpić również ograniczenie przyspieszenia środka koła. Ale jeżeli

$$\left(x, y\right) \in \text{Fr}\ \Omega, \quad \left(v_x, v_y\right) \in \text{Int}\ \mathsf{D}\Omega\left(x, y\right), \tag{7}$$

to wówczas ograniczenie takie nie powstaje, gdyż koło stykając się z brzegiem ograniczeń, oddala się od niego.

Ograniczenie przyspieszenia środka koła powstaje wtedy, gdy koło styka się z brzegiem ograniczenia, a prędkość środka koła należy do brzegu zbioru ograniczeń prędkości $\mathsf{D}\Omega$, czyli gdy

$$\left(x, y\right) \in \text{Fr}\ \Omega \quad \text{oraz} \quad \left(v_x, v_y\right) \in \text{Fr}\ \mathsf{D}\Omega\left(x, y\right). \tag{8a}$$

W takiej sytuacji koło stykając się z brzegiem ograniczenia może się po nim toczyć lub pozostaje nieruchome. Wówczas jest ograniczane przyspieszenie środka koła (a_x, a_y), gdyż powinna być spełniona relacja

$$\left(a_x, a_y\right) \in \mathsf{D}^2\,\Omega\left(\left(x, y\right), \left(v_x, v_y\right)\right) \tag{8b}$$

gdzie $\mathsf{D}^2\Omega$ jest odwzorowaniem nazywanym drugim następstwem relacji (2) – wyznaczającym zbiór dopuszczalnych przyspieszeń środka koła. W pracy [1] podano definicję tego odwzorowania, a szczegółowy opis dotyczący koła będzie zamieszczony w rozdziale 3.

W sytuacji określonej relacją (8a) działają więzy, to znaczy, że powstaje siła reakcji, której opis przedstawimy dalej. W czasie ruchu koła może również powstać sytuacja, gdy położenie i prędkość koła są takie, że:

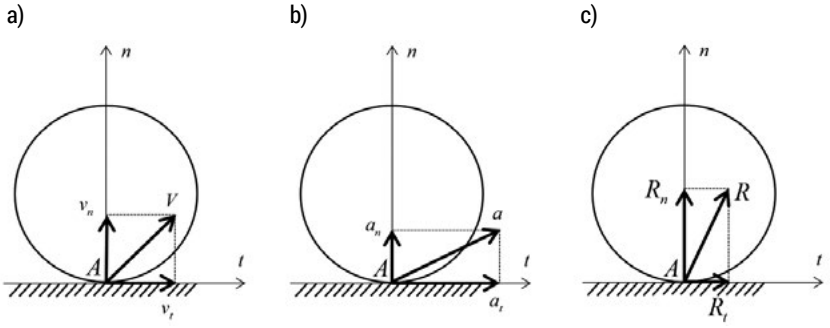
$$\left(x, y\right) \in \text{Fr}\ \Omega\,, \quad \left(v_x, v_y\right) \notin \mathsf{D}\Omega\left(x, y\right) \tag{9}$$

co oznacza, że prędkość koła jest niedopuszczalna przez więzy, gdyż nie spełnia warunku (6). W takim stanie następuje zderzenie koła ze sztywnymi ograniczeniami. W trakcie tego zderzenia prędkość koła zmienia się gwałtownie (skokowo) i osiąga wartość, która spełnia relację (6). Sformułowanie opisu zmiany prędkości w trakcie zderzenia będzie przedstawione dalej.

Jak już wspomniano, następstwa $D\Omega$ i $D^2\Omega$ określają zbiory dopuszczalnych prędkości i przyspieszeń koła, które styka się z powierzchnią ograniczającą (rys. 1). Zasady ustalania tych odwzorowań zostały szczegółowo opisane w pracach [1, 9]. Biorąc pod uwagę stosunkowo prostą postać brzegu zbioru Ω (rys. 2), ustalimy opisy tych odwzorowań w formie niewymagającej objaśnień.

3. Model oddziaływania więzów na koło

Oddziaływanie więzów na koło powstaje wtedy, gdy koło styka się z ograniczeniem. W takim położeniu może powstać siła reakcji działająca na koło, co szczegółowo omówimy dalej.



RYS. 4. Wektory prędkości, przyspieszenia i reakcji działające na koło stykające się z brzegiem zbioru Ω w punkcie A

FIG. 4. The velocity, acceleration and reaction vectors acting on a wheel in contact with the border of the set Ω at point A

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Na rysunku 4 pokazano przykładową ilustrację koła stykającego się z płaszczyzną ograniczającą jego ruch w punkcie *A*. Na rysunku 4a zaznaczono lokalny układ współrzędnych *Ant*, którego oś *An* jest prostopadła, a *At* – styczna do brzegu Fr Ω; zaznaczono również prędkość punktu *A* leżącego na obwodzie koła. Ograniczenia ruchu koła powodują, że składowa normalna tej prędkości powinna spełniać warunek:

$$v_n \geq 0 \tag{10}$$

Prędkość, dla której $v_n < 0$ jest niedopuszczalna przez więzy; w takiej sytuacji powstaje zderzenie, które będzie rozważane dalej.

Jeżeli $v_n > 0$, to więzy nie ograniczają ruchu, gdyż koło oddala się od brzegu.

Na rysunku 4b pokazano wektor przyspieszenia punktu A.

Jeżeli prędkość punktu A jest taka, że $v_n = 0$, to wtedy więzy ograniczają to przyspieszenie

$$a_n \geq 0 \quad \text{gdy} \quad v_n = 0 \tag{11}$$

W opisanej wyżej sytuacji powstaje siła reakcji, którą zilustrowano na rysunku 4c. W przypadku więzów idealnych działa tylko siła docisku $R_n \geq 0$, natomiast składowa styczna nie powstaje, czyli $R_t = 0$.

Dla więzów nieidealnych przyjmuje się hipotezę, według której składowa styczna odwzorowuje siłę tarcia między kołem i brzegiem więzów. Tak określony model więzów nieidealnych opisujemy relacjami, które wyznaczają związek pomiędzy siłą reakcji R a przyspieszeniem i prędkością punktu A należącego do koła:

- dla składowej normalnej R_n mamy:

$$R_n \geq 0, \quad a_n \geq 0, \quad a_n R_n = 0, \quad \text{gdy} \quad v_n = 0 \tag{12a}$$

- składowa styczna R_t, gdy $R_n > 0$ jest określona relacjami:

$$R_t = \mu R_n \tau, \quad \tau := \operatorname{sign} v_t, \quad \text{gdy} \quad v_t \neq 0 \tag{12b}$$

$$R_t = \mu R_n \tau, \quad \tau \in \left\{ \begin{matrix} \{\operatorname{sign} a_t\} & \text{gdy} & a_t \neq 0 \\ [-1,+1] & \text{gdy} & a_t = 0 \end{matrix} \right\} \quad \text{gdy} \quad v_t = 0 \tag{12c}$$

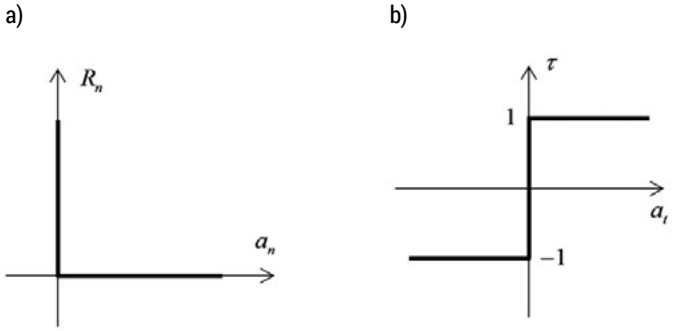
gdzie μ – współczynnik tarcia między kołem i brzegiem więzów; τ – mnożnik tarcia.

Te relacje można zilustrować wykresami; na rysunku 5a zamieszczono wykres relacji (12a), natomiast relację (12c) zilustrowano na rysunku 5b. Poza tym relacje (12a) (rys. 5a) można równoważnie opisać zależnością

$$R_n = \left[R_n - \rho a_n \right]^+, \quad \rho > 0 \tag{13a}$$

gdzie $[\cdot]^+$ oznacza funkcję taką, że:

$$\left[\xi \right]^+ := \begin{cases} \xi & \text{gdy} \quad \xi \geq 0 \\ 0 & \text{gdy} \quad \xi < 0 \end{cases} \tag{13b}$$



RYS. 5. Wykresy relacji (12a) i (12c)
FIG. 5. The relation graphs (12a) i (12c)

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Analogicznie, w przypadku relacji (11c) (rys. 5b) otrzymujemy

$$\tau = \Pi\left(\tau + \rho\, a_t\right), \quad \rho > 0 \tag{14a}$$

gdzie funkcję Π zdefiniowano następująco:

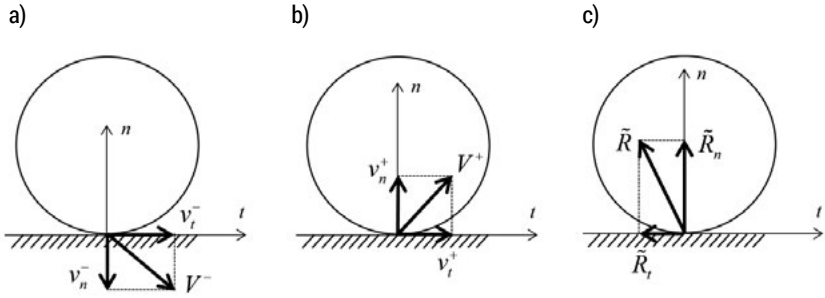
$$\Pi\left(\xi\right) := \begin{cases} \xi & \text{gdy} \quad |\xi| \le 1 \\ \operatorname{sign}\xi & \text{gdy} \quad |\xi| > 1 \end{cases} \tag{14b}$$

W przedstawionej powyżej hipotezie, dotyczącej wzajemnego oddziaływania stykających się nieodkształcalnych ciał, nie uwzględniliśmy momentu sił nazywanego *oporami toczenia się koła*. Jest to moment tarcia, którego graniczna wartość jest proporcjonalna do siły nacisku R_n. Moment ten jest przeciwnie skierowany do prędkości kątowej koła. W celu uproszczenia opisu reakcji działającej na koło, a także zakładając bardzo małą wartość oporów toczenia, pominięto ich wpływ na ruch koła.

Teraz rozpatrzymy sytuację, w której koło zderza się z więzami, czyli gdy $v_n < 0$. Schematyczny opis prędkości koła tuż przed zderzeniem V^- oraz po zderzeniu V^+ pokazano na rysunkach 6a i 6b, natomiast na rysunku 6c zaznaczono impulsy siły reakcji $\tilde{R}_n$ i $\tilde{R}_t$, wywołujące zmianę prędkości koła. Impuls cierny $\tilde{R}_t$ ustalamy według hipotezy Routha [3].

Wspomniany efekt zderzenia opisuje się relacją między prędkościami po zderzeniu V^+ (rys. 6b) a impulsami reakcyjnymi (rys. 6c). Postać tej relacji formułuje się stosownie do przyjętej hipotezy. Na przykład, gdy analizuje się więzy gładkie (doskonałe), to pomija się składową styczną, czyli $\tilde{R}_t = 0$.

W niniejszej pracy przedstawimy opis hipotezy o zderzeniu plastycznym, a także zderzeniu sprężystym lub sprężysto-plastycznym, przy założeniu, że więzy są niedoskonałe.



RYS. 6. **Prędkości i impulsy działające na koło zderzające się z więzami**
FIG. 6. **The velocities and impulses acting on a wheel colliding with ties**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Opis zderzenia plastycznego ma postać analogiczną do opisu siły reakcji podanego we wzorach (12). Zderzenie plastyczne jest określone relacjami:

$$\tilde{R}_n \geq 0, \quad v_n^+ \geq 0, \quad v_n^+ \tilde{R}_n = 0 \tag{15a}$$

$$\tilde{R}_t = \mu \, \tilde{R}_n \tau \, , \quad \tau \in \begin{cases} \left\{ \text{sign } v_t^+ \right\} & \text{gdy} \quad v_t^+ \neq 0 \\ \left[-1, +1 \right] & \text{gdy} \quad v_t^+ = 0 \end{cases} \tag{15b}$$

Relację (15b) ustalono na podstawie hipotezy Routha [3].

Wykresy ilustrujące te relacje są analogiczne do pokazanych na rysunku 5 i można je przedstawić w równoważnej postaci, analogicznie do wzorów (13) i (14).

Opis zderzenia sprężystego lub sprężysto-plastycznego opiera się na hipotezie Newtona-Poissona analizowanej w pracy [1]. Według tej hipotezy wypadkowy impuls siły reakcji w trakcie takiego zderzenia określają wzory:

$$\tilde{R}_n^* = \left(1 + \beta \right) \tilde{R}_n \tag{16a}$$

$$\tilde{R}_t^* = \mu \, \tilde{R}_n^* \tau^* \tag{16b}$$

$$\tau^* \in \begin{cases} \left\{ \text{sign } v_t^+ \right\} & \text{gdy} \quad v_t^+ \neq 0 \\ \left[-1, +1 \right] & \text{gdy} \quad v_t^+ = 0 \end{cases} \tag{16c}$$

gdzie $\tilde{R}_n$, $\tilde{R}_t$ – impulsy opisane we wzorach (15); $\beta \in \left(0, 1\right]$ – współczynnik restytucji, którego wartość dla zderzenia sprężystego wynosi 1.

4. Opis ruchu koła

Opis ruchu koła przedstawimy w postaci macierzowej przy użyciu współrzędnych uogólnionych. W tym celu wprowadzamy następujące oznaczenia:

$X := \begin{bmatrix} x \\ y \\ \varphi \end{bmatrix}$ – wektor współrzędnych uogólnionych,

$M := \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix}$ – macierz bezwładności koła,

$Q := \begin{bmatrix} 0 \\ -mg \\ 0 \end{bmatrix}$ – wektor siły grawitacji działającej na koło,

$F_n \in R^3$ – wektor siły reakcji prostopadły do brzegu zbioru Ω,

$F_t \in R^3$ – wektor siły reakcji styczny do brzegu zbioru Ω,

$X \in \Omega \subset R^3$ – relacja opisująca ograniczenie ruchu koła.

Jeżeli położenie koła jest takie, że $X \in$ Int Ω, to wtedy więzy nie działają, a opis ruchu koła ma postać:

$$M\ddot{X} = Q \tag{17a}$$

a jeżeli ruch koła jest ograniczany, co może nastąpić, gdy $X \in$ Fr Ω, to wtedy w równaniu ruchu należy uwzględnić opis sił reakcji:

$$M\ddot{X} = Q + F_n - F_t \tag{17b}$$

W następnym podrozdziale ustalimy metodę wyznaczania sił reakcji.

Wyznaczanie sił reakcji

W przedstawionych dalej opisach ograniczeń oraz sił reakcji będą stosowane funkcje wektorowe G i H, które każdemu wektorowi X spełniającemu relację $X \in$ Fr Ω przyporządkowują parę wektorów: prostopadły $G \in R^3$ oraz styczny $H \in R^3$. Za pomocą tych wektorów można wyznaczyć prędkości i przyspieszenie punktu koła stykającego się z brzegiem zbioru Ω (patrz rys. 3):

$$v_n := G^T(X)\dot{X}, \quad v_t := H^T(X)\dot{X} \tag{18a}$$

$$a_n := G^T(X)\ddot{X} + \gamma_n\left(X, \dot{X}\right), \quad a_t := H^T(X)\ddot{X} + \gamma_t\left(X, \dot{X}\right) \tag{18b}$$

gdzie γ_n i γ_t są funkcjami skalarnymi. Szczegółowe postacie funkcji wektorowych G i H oraz skalarnych γ_n i γ_t będą podane dalej.

Przy użyciu funkcji G oraz γ_n można opisać następstwo $\mathsf{D}\Omega$, według wzoru (6), wyznaczające zbiór dopuszczalnych prędkości $\dot{X}$:

$$\mathsf{D}\Omega(X) := \left\{\dot{X} \in R^3 : \quad G^T(X)\dot{X} \geq 0\right\}, \text{ gdy } X \in \operatorname{Fr}\Omega \tag{19a}$$

a także zbiór dopuszczalnych przyspieszeń według wzoru (8b):

$$\mathsf{D}^2\Omega\left(X, \dot{X}\right) := \left\{\ddot{X} \in R^3 : \quad G^T(X)\ddot{X} + \gamma_n\left(X, \dot{X}\right) \geq 0\right\}, \; X \in \operatorname{Fr}\Omega \text{ i } G^T(X)\dot{X} = 0 \tag{19b}$$

Jeżeli położenie X i prędkość $\dot{X}$ spełniają relacje wymienione we wzorze (19b), to wtedy powstaje siła reakcji F, której dwie składowe F_n i F_t występują w równaniu ruchu (17b).

Uogólnione siły reakcji F_n, $F_t \in R^3$ opiszemy na podstawie relacji, które podano we wzorach (12).

Siłę reakcji normalnej wyznacza wzór:

$$F_n := G(X)\lambda \tag{20a}$$

jeśli $\lambda \geq 0$ jest mnożnikiem reakcji spełniającym relacje wynikające ze wzoru (12a), czyli:

$$\lambda \geq 0, \quad G^T(X)\ddot{X} + \gamma_n\left(X, \dot{X}\right) \geq 0, \quad \lambda\left(G^T(X)\ddot{X} + \gamma_n\left(X, \dot{X}\right)\right) = 0 \tag{20b}$$

Jeżeli uwzględnimy opis relacji ze wzoru (11), to otrzymamy opis równoważny:

$$\lambda = \left[\lambda - \rho\left(G^T(X)\ddot{X} + \gamma_n\left(X, \dot{X}\right)\right)\right]^+, \text{ gdy } X \in \operatorname{Fr}\Omega \text{ oraz } G^T(X)\dot{X} = 0 \tag{21}$$

jeśli ρ jest dowolną liczbą dodatnią.

Podobnie, według wzorów (12b) i (12c), wyznaczamy relacje opisujące reakcję styczną $F_t \in R^3$:

$$F_t = \mu\, H(X)\|G\|\lambda\tau \tag{22a}$$

$$\tau = \operatorname{sign} v_t \equiv \operatorname{sign}\left(H^T(X)\dot{X}\right), \text{ gdy } v_t \neq 0 \tag{22b}$$

$$\tau \in \begin{Bmatrix} \{\operatorname{sign} a_t\} & \text{gdy} & a_t \neq 0 \\ [-1, +1] & \text{gdy} & a_t = 0 \end{Bmatrix} \text{ gdy } v_t = 0 \tag{22c}$$

jeśli wektor a_t opisano we wzorach (18b).

We wzorze (20a) uwzględniono fakt, że nacisk koła na brzeg ograniczeń, określony siłą ze wzoru (20a), wynosi:

$$N := \|F_n\| = \|G\|\lambda \tag{23a}$$

a wynikająca stąd graniczna wartość siły tarcia (wg Coulomba) jest równa:

$$T_o := \mu N = \mu \|G\|\lambda \tag{23b}$$

Relację między mnożnikiem τ i przyspieszeniem $\ddot{X}$ daną wzorem (22c) można, analogicznie do relacji (21), przedstawić w postaci:

$$\tau = \Pi\left(\tau + \rho\left(H^T(X)\ddot{X} + \gamma_t(X,\dot{X})\right)\right), \text{ gdy } \lambda > 0 \tag{23c}$$

gdzie ρ jest dowolną liczbą dodatnią.

Przyspieszenie koła

Równania ruchu przedstawiają zestaw relacji, na podstawie których w każdej chwili $t \in \left[0, t_{end}\right)$ można wyznaczyć wektor przyspieszenia koła $\ddot{X}(t) \in R^3$, pod warunkiem, że są spełnione ograniczenia ruchu, czyli gdy $X(t) \in \Omega$ oraz $\dot{X}(t) \in \mathrm{D}\Omega\left(X(t)\right)$.

Jeżeli koło styka się z brzegiem ograniczenia, czyli gdy

$$X(t) \in \mathrm{Fr}\,\Omega \text{ oraz } \dot{X}(t) \in \mathrm{Fr}\,\mathrm{D}\Omega\left(X(t)\right) \tag{24}$$

to na koło działają siły reakcji więzów rozpatrywane w poprzednim podrozdziale.

Na tej podstawie formułujemy zestaw relacji, z których można jednoznacznie wyznaczyć wartość wektora przyspieszenia. Zestaw ten zawiera:

- równanie ruchu (17b), które jest napisane z uwzględnieniem wzorów (20a) i (22a):

$$M\ddot{X} = Q + G\lambda - \mu\, H\|G\|\lambda\tau \tag{25a}$$

- relacje określające mnożniki według wzorów (21) i (24):

$$\lambda = \left[\lambda - \rho_1\left(G^T\ddot{X} + \gamma_n\right)\right]^+, \quad \rho_1 > 0 \tag{25b}$$

$$\tau = \begin{cases} \mathrm{sign}\left(H^T\dot{X}\right) & \text{gdy} \quad H^T\dot{X} \neq 0 \\ \Pi\left(\tau + \rho_2\left(H^T\ddot{X} + \gamma_t\right)\right), \quad \rho_2 > 0 & \text{gdy} \quad H^T\dot{X} = 0 \end{cases} \tag{25c}$$

przy czym we wzorach (25) nie wymieniono argumentów odwzorowań G, H, γ_n, γ_t.

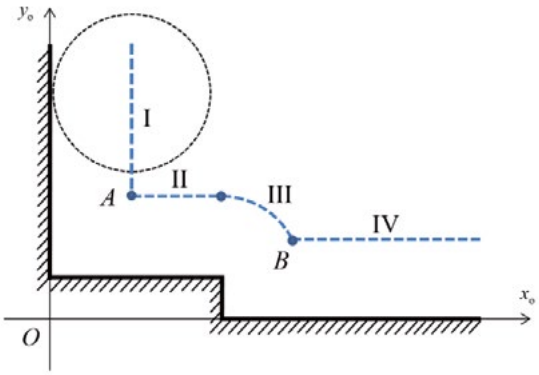
Równania (25) wyznaczają w każdej chwili $t \in [0, t_{end})$, wartość wektora przyspieszenia $\ddot{X}(t)$ oraz mnożników $\lambda(t)$ i $\tau(t)$, które określają wartości sił reakcji.

W relacjach (25) występują odwzorowania G, H, γ_n i γ_t, których opis przedstawimy w następnym podrozdziale.

Opis dodatkowych odwzorowań

Wymienione w poprzednim podrozdziale odwzorowania G, H, γ_n i γ_t służą do opisu geometrycznych cech brzegu ograniczeń $\mathrm{Fr}\Omega$ przy użyciu współrzędnych uogólnionych. Jak już wspomniano, odwzorowanie wektorowe G wyznacza, w każdym punkcie brzegu $X \in \mathrm{Fr}\Omega$, wektor $G(X) \in R^3$, który jest prostopadły, a odwzorowanie H wyznacza wektor $H(X)$, który jest styczny do brzegu ograniczeń. Odwzorowania skalarne γ_n i γ_t służą do wyznaczenia wpływu krzywizny brzegu oraz prędkości koła na jego przyspieszenie.

Rozpatrywany brzeg zbioru przedstawiono na rysuku 2; jest to krzywa złożona z czterech segmentów, zawierająca dwa punkty osobliwe A i B (por. rys. 7).



RYS. 7. Brzeg zbioru Fr Ω z zaznaczonymi segmentami i punktami osobliwymi
FIG. 7. The boundary of the set with selected segments and singular points

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Opis rozpatrywanych odwzorowań zależy od segmentu krzywej i wynika ze wzorów (3) i (4). Stąd otrzymujemy:

- segment I, $\mathrm{Fr}^{\mathrm{I}}\,\Omega := \left\{(x,y):\quad x = r,\ \ y > r + h\right\}$

$$G_{\mathrm{I}} := \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad H_{\mathrm{I}} := \begin{bmatrix} 0 \\ 1 \\ -r \end{bmatrix}, \quad \gamma_n^{\mathrm{I}} = \gamma_t^{\mathrm{I}} = 0 \tag{26a}$$

- segment II, $\mathrm{Fr}^{\mathrm{II}}\,\Omega := \left\{(x,y):\quad r < x \le l_1,\ \ y = r + h\right\}$

$$G_{\mathrm{II}} := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad H_{\mathrm{II}} := \begin{bmatrix} 1 \\ 0 \\ r \end{bmatrix}, \quad \gamma_n^{\mathrm{II}} = \gamma_t^{\mathrm{II}} = 0 \tag{26b}$$

- segment III, $\mathrm{Fr}^{\mathrm{III}}\,\Omega := \left\{(x,y):\ \ l_1 < x < l_1 + a,\ \ y = f(x)\right\}$

$$G_{\mathrm{III}} := \begin{bmatrix} -f'(x) \\ 1 \\ 0 \end{bmatrix},\quad H_{\mathrm{III}} := \begin{bmatrix} 1 \\ f'(x) \\ r \end{bmatrix},\quad \gamma_n^{\mathrm{III}} := f''(x)\dot{x}^2,\quad \gamma_t^{\mathrm{III}} := f''(x)\dot{x}\dot{y} \tag{26c}$$

- segment IV, $\mathrm{Fr}^{\mathrm{IV}}\,\Omega := \left\{(x,y):\ \ x > l_1 + a,\ \ y = r\right\}$

$$G_{\mathrm{IV}} := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},\quad H_{\mathrm{IV}} := \begin{bmatrix} 1 \\ 0 \\ r \end{bmatrix},\quad \gamma_n^{\mathrm{IV}} = \gamma_t^{\mathrm{IV}} = 0 \tag{26d}$$

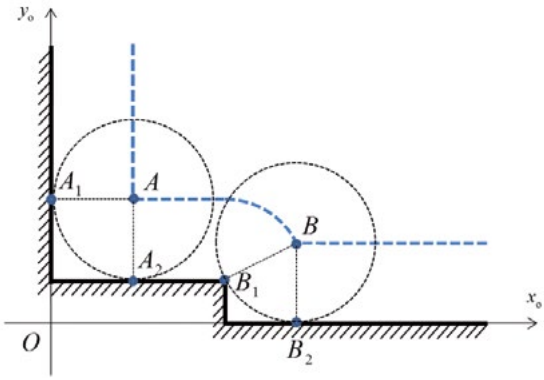
We wzorach (26) użyto oznaczeń zaznaczonych na rysunku 2 oraz związanych z funkcją *f*, opisaną we wzorze (4c); stąd otrzymuje się:

$$\left.\begin{aligned} f'(x) &:= -\frac{x - l_1}{\sqrt{r^2 - (x - l_1)^2}} \\ f''(x) &:= -\frac{r^2}{\left[r^2 - (x - l_1)^2\right]^{3/2}} \end{aligned}\right\} \quad \text{gdy} \quad l_1 < x \le l_1 + a \tag{26e}$$

Osobliwe położenia koła są określone współrzędnymi środka koła, tzn.:

$$\text{punkt } A: \quad x = r,\ \ y = r + h \tag{27a}$$

$$\text{punkt } B: \quad x = l_1 + a,\ \ y = r \tag{27b}$$



RYS. 8. **Osobliwe położenia, w których koło styka się z ograniczeniami w dwóch punktach**
FIG. 8. **The singular positions in which the wheel is in contact with the constraints in two points**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

W tych położeniach koło styka się z ograniczeniami w dwóch punktach (rys. 8). Stąd wynikają następujące opisy zbiorów dopuszczalnych prędkości

$$\mathrm{D}\Omega\left(X_A\right) := \left\{\dot{X} \in R^3 : \quad G_{\mathrm{I}}^T \dot{X} \geq 0, G_{\mathrm{II}}^T \dot{X} \geq 0\right\} \tag{28a}$$

$$\mathrm{D}\Omega\left(X_B\right) := \left\{\dot{X} \in R^3 : \quad G_{\mathrm{II}}^T \dot{X} \geq 0, G_{\mathrm{III}}^T \dot{X} \geq 0\right\} \tag{28b}$$

gdzie użyto oznaczeń ze wzorów (26).

Jak już wcześniej wspomniano, ograniczenie przyspieszenia koła oraz siła reakcji powstają wtedy, gdy wektor prędkości znajduje się na brzegu zbioru $\mathrm{D}\Omega$, co zostało opisane we wzorach (19) i (20).

Na tej podstawie określamy relacje dla normalnej składowej siły reakcji w położeniu A (27a):

$$F_n^A := \begin{cases} G_{\mathrm{I}}\lambda_{\mathrm{I}} & \text{gdy} \quad G_{\mathrm{I}}^T \dot{X} = 0, G_{\mathrm{II}}^T \dot{X} > 0 \quad \text{dla punktu } A_1 \\ G_{\mathrm{II}}\lambda_{\mathrm{II}} & \text{gdy} \quad G_{\mathrm{I}}^T \dot{X} > 0, G_{\mathrm{II}}^T \dot{X} = 0 \quad \text{dla punktu } A_2 \\ G_{\mathrm{I}}\lambda_{\mathrm{I}} + G_{\mathrm{II}}\lambda_{\mathrm{II}} & \text{gdy} \quad G_{\mathrm{I}}^T \dot{X} = 0, G_{\mathrm{II}}^T \dot{X} = 0 \quad \text{dla obu punktów} \end{cases} \tag{29}$$

Wartości mnożników λ_{I} i λ_{II} są związane z wektorem przyspieszenia $\ddot{X}$ relacją opisaną we wzorze (25b).

Siłę reakcji F_n^B, działającą na koło w położeniu B (27b), obliczamy analogicznie do wzoru (29), czyli:

$$F_n^B := \begin{cases} G_{\mathrm{III}}\lambda_{\mathrm{III}} & \text{gdy} \quad G_{\mathrm{III}}^T \dot{X} = 0, G_{\mathrm{IV}}^T \dot{X} > 0 \quad \text{dla punktu } B_1 \\ G_{\mathrm{IV}}\lambda_{\mathrm{IV}} & \text{gdy} \quad G_{\mathrm{III}}^T \dot{X} > 0, G_{\mathrm{IV}}^T \dot{X} = 0 \quad \text{dla punktu } B_2 \\ G_{\mathrm{III}}\lambda_{\mathrm{III}} + G_{\mathrm{IV}}\lambda_{\mathrm{IV}} & \text{gdy} \quad G_{\mathrm{III}}^T \dot{X} = 0, G_{\mathrm{IV}}^T \dot{X} = 0 \quad \text{dla obu punktów} \end{cases} \tag{30}$$

Postać opisu relacji między mnożnikami λ i wektorem przyspieszenia $\ddot{X}$ określono wzorem (25b).

Styczną składową siły reakcji F_t określimy na podstawie relacji (22), ale po uwzględnieniu właściwego indeksu przy wektorach G, H i mnożnikach λ, τ.

5. Opis zderzenia

W sytuacji określonej relacją (9) prędkość koła jest niedopuszczalna przez więzy, co skutkuje zderzeniem, w czego efekcie powstaje impulsowa reakcja więzów, dostosowująca prędkość koła do ograniczeń. W rozdziale 3 przedstawiono lokalny opis

modelu zderzenia w postaci relacji określającej związek między reakcją impulsową i skokową zmianą prędkości koła. Zamieszczone tam relacje przedstawimy teraz przy użyciu współrzędnych uogólnionych.

Opis zderzenia plastycznego obejmuje równanie bilansu pędu oraz relacje określające impulsowe siły reakcji. Ogólna postać tego opisu jest analogiczna do równań ruchu (17) i (25), czyli:

$$M\dot{X}^{+} - M\dot{X}^{-} = \tilde{F}_n - \tilde{F}_t \tag{31}$$

$$\tilde{F}_n = G\tilde{\lambda}, \quad \tilde{F}_t = \mu \left\| G \right\| H\tilde{\lambda}\tau \tag{32}$$

a opis relacji ze wzorów (15) ma teraz postać analogiczną do (25b) i (25c), czyli:

$$\tilde{\lambda} = \left[\tilde{\lambda} - \rho_1 G^T \dot{X}^{+} \right]^{+} \tag{33a}$$

$$\tau = \Pi\left(\tau + \rho_2 H^T \dot{X}^{+}\right) \tag{33b}$$

gdzie indeksem „~" zaznaczono impulsy siły działającej na koło.

Gdy zderzenie koła z więzami powstaje w osobliwym położeniu koła, określanym punktem *A* lub *B* (według wzorów (27)), to wtedy na koło działają dwie reakcje, czyli:

$$\tilde{F}_n = G_{\mathrm{I}}\tilde{\lambda}_{\mathrm{I}} + G_{\mathrm{II}}\tilde{\lambda}_{\mathrm{II}} \text{ dla punktu } A \tag{34a}$$

$$\tilde{F}_n = G_{\mathrm{III}}\tilde{\lambda}_{\mathrm{III}} + G_{\mathrm{IV}}\tilde{\lambda}_{\mathrm{IV}} \text{ dla punktu } B \tag{34b}$$

Relacje określające mnożniki $\tilde{\lambda}$ i τ ze wzorów (34) mają postać taką jak we wzorach (33), ale po uwzględnieniu stosownych indeksów.

Jak już wspomniano, relacje ze wzorów (32) i (33) określają skokową zmianę prędkości koła $\Delta V := V^{+} - V^{-}$ oraz impulsy siły reakcji $\tilde{F}_n$ i $\tilde{F}_t$ powstające podczas zderzenia plastycznego.

Jeżeli rozważa się zderzenie sprężyste lub sprężysto-plastyczne, to – zgodnie z hipotezą Newtona-Poissona określoną we wzorach (16) – należy uwzględnić dodatkową fazę zderzenia (restytucji) opisaną równaniami:

$$M\dot{X}^{*} - M\dot{X}^{+} = \tilde{F}_n^{*} - \tilde{F}_t^{*} \tag{35a}$$

gdzie $\dot{X}^{*} \in R^3$ – wektor prędkości po zderzeniu sprężystym lub sprężysto-plastycznym; F_n^{*}, F_t^{*} – impulsy sił reakcji powstające w tej fazie, określone relacjami:

$$F_n^{*} = G\lambda^{*}, \ \lambda^{*} = \beta\,\tilde{\lambda} \tag{35b}$$

$$F_t^{*} = \mu \left\| G \right\| H\lambda^{*}\tau^{*}, \ \tau^{*} = \Pi\left(\tau^{*} + \rho_2 H^T \dot{X}^{*}\right) \tag{35c}$$

W wymienionych relacjach wartość mnożnika λ^{*} jest znana, co wynika ze wzoru (35c), gdzie $\beta \in (0, 1]$ jest współczynnikiem restytucji, którego wartość dla zderzenia sprężystego wynosi 1.

Relacje (35) wyznaczają wektor prędkości po zderzeniu $\dot{X}^*$ oraz mnożnik τ^* określający wpływ tarcia na tę fazę zderzenia.

6. Symulacja ruchu koła

Opis metody rozwiązania zagadnienia początkowego

Niegładkie zagadnienie ruchu układu mechanicznego było szczegółowo analizowane w monografii [3], gdzie rozpatrywano jednostronne więzy doskonałe. Z przedstawionych tam rozważań wynika, że ruch takiego układu opisuje funkcja absolutnie ciągła. Określono również etapową metodę wyznaczania takiej funkcji. Metody tej użyjemy do wyznaczenia funkcji opisującej ruch koła.

Rozpatrywane w rozdziałach 4 i 5 opisy relacji, określające ruch koła, przedstawimy teraz w syntetycznej postaci za pomocą dwóch równań

$$\ddot{X} = \mathsf{A}\left(X, \dot{X}\right), \quad \text{gdy} \quad X \in \Omega, \ \dot{X} \in \mathsf{D}\Omega\left(X\right) \tag{36}$$

$$\dot{X}^+ = \mathsf{Z}\left(\dot{X}^-, X\right), \quad \text{gdy} \quad X \in \mathrm{Fr}\,\Omega, \ \dot{X}^- \notin \mathsf{D}\Omega\left(X\right) \tag{37}$$

gdzie $X \in R^3$ – wektor współrzędnych uogólnionych, $\mathsf{A} : R^6 \to R^3$ – odwzorowanie wyznaczające wektor przyspieszenia, a $\mathsf{Z} : R^3 \to R^3$ – odwzorowanie wyznaczające skokową zmianę prędkości w chwilach zderzenia.

Odwzorowanie A jest określone w postaci uwikłanej za pomocą relacji (25). W tym przypadku ustalenie wartości odwzorowania A sprowadza się do wyznaczenia rozwiązania relacji (25), czyli obliczenia wartości mnożników λ i τ oraz wektora $\ddot{X}$.

Odwzorowanie Z jest również uwikłane, a jego wartości wyznacza rozwiązanie relacji (32)÷(35), określone mnożnikami $\tilde{\lambda}$ i τ oraz wektorem $\dot{X}^*$.

W każdym etapie wspomnianej wyżej metody wyznacza się rozwiązanie zagadnienia początkowego określonego wzorem (36). Tak ustalona funkcja X, która jest ciągła i różniczkowalna, opisuje ruch koła do chwili, w której powstaje zderzenie określone warunkami ze wzoru (37). Wówczas następuje skokowa zmiana prędkości opisana odwzorowaniem Z; w tej sytuacji kończy się rozpatrywany etap rozwiązania.

Według tej etapowej metody wyznaczania rozwiązania zagadnienia początkowego opracowano program obliczeń komputerowych służący do symulacji ruchu koła. Program ten zawiera algorytmy wyznaczania wartości niejawnych odwzorowań A i Z we wszystkich segmentach zbioru $\mathrm{Fr}\Omega$ opisanych wzorami (26) oraz obu punktach osobliwych (27) zaznaczonych na rysunku 8. Należy dodać, że aby ustalić wartości odwzorowania niejawnego A, trzeba rozwiązać układ równań algebraicznych zawierający pięć niewiadomych: $\ddot{X} \in R^3$, λ i τ. Analogicznie wyznacza się wartości niejawnego odwzorowania Z.

Szczegółowa postać równań ruchu i reakcji

Na początku rozważymy szczegółową postać równań ruchu i reakcji, w której koło styka się z brzegiem w sektorze IV (patrz rys. 7). Na rysunku 3 koło z indeksem 6 ilustruje takie położenie. W rozpatrywanej sytuacji są spełnione warunki określone we wzorze (24), czyli $x > l_1 + a$, $y = r$ oraz $\dot{y} = 0$, gdzie a, l_1 – wymiary opisane we wzorze (4b).

Równania ruchu (25), w rozwiniętej postaci oraz po uwzględnieniu wzorów (26d), dla sektora IV, mają postać:

$$m\ddot{x} = -\mu\,\lambda\tau \tag{38}$$

$$m\ddot{y} = -mg + \lambda \tag{39}$$

$$J\ddot{\varphi} = -\mu\,\lambda\, r\tau \tag{40}$$

$$\lambda = \left[\lambda - \rho\,\ddot{y}\right]^{+}, \quad \rho > 0 \tag{41}$$

$$\tau = \begin{cases} \operatorname{sign}\left(\dot{x} + r\dot{\varphi}\right) & \text{gdy} \quad \dot{x} + r\dot{\varphi} \neq 0 \\ \Pi\left(\tau + \rho\left(\ddot{x} + r\ddot{\varphi}\right)\right), \ \rho > 0 & \text{gdy} \quad \dot{x} + r\dot{\varphi} = 0 \end{cases} \tag{42}$$

Powyższy układ równań opisuje przyspieszenia koła ($\ddot{x}$, $\ddot{y}$, $\ddot{\varphi}$) oraz mnożniki reakcji (λ, τ). Jest to uwikłana postać odwzorowania A, w której koło styka się z brzegiem sektora IV. Wyznaczenie rozwiązania powyższych równań jest stosunkowo proste:

- jeżeli koło ślizga się, czyli $\dot{x} + r\dot{\varphi} \neq 0$, to z równania (42_1) wyznacza się wprost wartość τ;
- jeżeli koło toczy się bez poślizgu, to z równań (42_2) oraz (38) i (39) otrzymujemy, że $\tau = 0$;
- ze stosunkowo prostego układu równań (39) i (41) otrzymujemy $\lambda = mg$;
- po ustaleniu wartości λ i τ można z równań (38), (39) i (40) wyznaczyć wartości przyspieszeń, a w szczególności $\ddot{y} = 0$.

Opisane równania (38-42) dotyczą ruchu koła poruszającego się (toczącego się) po brzegu. Teraz zajmiemy się opisem zderzenia koła z brzegiem w sektorze IV. Ogólny opis zderzenia zamieszczono we wzorach (32), (33) oraz (35). W rozpatrywanej sytuacji rozwinięty opis zderzenia, przy założeniu hipotezy zderzenia plastycznego, ma postać:

$$m\dot{x}^{+} - m\dot{x}^{-} = -\mu\,\tilde{\lambda}\tau \tag{43}$$

$$m\dot{y}^{+} - m\dot{y}^{-} = \tilde{\lambda} \tag{44}$$

$$J\dot{\varphi}^{+} - J\dot{\varphi}^{-} = -\mu\,\tilde{\lambda}\, r\tau \tag{45}$$

$$\tilde{\lambda} = \left[\tilde{\lambda} - \rho\,\dot{x}^{+}\right]^{+}, \quad \rho > 0 \tag{46}$$

$$\tau = \Pi\left(\tau + \rho v_t^+\right), \quad \rho > 0 \tag{47}$$

$$v_t^+ := \dot{x}^+ + r\dot{\varphi}^+ \tag{48}$$

Jest to układ równań względem $\dot{x}^+$, $\dot{y}^+$, $\tilde{\lambda}$, τ. Wyznaczenie rozwiązania uzyskujemy w następujących krokach:

- z równań (44) i (46) można bezpośrednio wyznaczyć $\tilde{\lambda} = -m\dot{y}^-$;
- według wzoru (48) oraz równań (43) i (44) otrzymujemy zależność:

$$v_t^+ := v_t^- - \left(\frac{1}{m} + \frac{r^2}{J}\right)\mu\,\tilde{\lambda}\tau\,; \quad v_t^- := \dot{x}^- + r\dot{\varphi}^- \tag{49}$$

następnie, z równania (47), wyznaczamy wartości τ oraz v_t^+;

- po ustaleniu wartości $\tilde{\lambda}$ i τ obliczamy prędkości po zderzeniu $\dot{x}^+$ oraz $\dot{y}^+$.

Wyznaczone rozwiązanie określa efekt zderzenia plastycznego. Jeżeli rozpatruje się zderzenie sprężyste albo sprężysto-plastyczne, to należy uwzględnić zmianę prędkości w drugiej fazie zderzenia, opisanej we wzorach (35). W rozpatrywanym przykładzie opis ten, w odniesieniu do współczynnika restytucji $\beta \in (0,1]$, ma postać:

$$\tilde{\lambda}^* = \beta\,\tilde{\lambda} \quad \tilde{\lambda}^* = \beta\tilde{\lambda} \tag{50}$$

$$m\dot{x}^* = m\dot{x}^+ - \mu\,\tilde{\lambda}^*\tau^* \tag{51}$$

$$m\dot{y}^* = m\dot{y}^+ + \tilde{\lambda}^* \tag{52}$$

$$J\dot{\varphi}^* = J\dot{\varphi}^+ - \mu\,\tilde{\lambda}^* r\tau^* \tag{53}$$

$$\tau^* = \Pi\left(\tau^* + \rho v_t^*\right), \quad \rho > 0 \tag{54}$$

$$v_t^* := \dot{x}^* + r\dot{\varphi}^* = v_t^+ - \left(\frac{1}{m} + \frac{r^2}{J}\right)\mu\,\tilde{\lambda}^*\tau^* \tag{55}$$

Efekt zderzenia opisują prędkości $\dot{x}^*$, $\dot{y}^*$, $\dot{\varphi}^*$ oraz mnożniki λ^* i τ^*. Wartość λ^* jest określona wzorem (50), natomiast z równań (54) i (55) otrzymuje się bezpośrednio wartości τ^* i v_t^+.

Podczas ruchu koła w sektorze III (por. rys. 7), koło styka się tylko z narożnikiem krawężnika, co ilustruje rysunek 3 (koło z indeksem nr 4). W tym przypadku środek koła zajmuje położenie określone warunkami (patrz wzór (26c)):

$$l_1 \le x < l_1 + a,\ y = f(x) \tag{56a}$$

a prędkość koła jest taka, że

$$v_n := \dot{y} - f'(x)\dot{x} = 0 \tag{56b}$$

Warunek (56b) oznacza, że normalna składowa prędkości punktu koła (rys. 3 oraz wzór (10)) stykającego się z więzami jest równa zero, czyli koło nie odrywa się od krawężnika.

We wzorach (26c) zamieszczono definicje wektorów G i H oraz wyrażeń γ_n i γ_t. W celu uproszczenia dalszych opisów pominiemy indeksy „III" znajdujące się przy wymienionych oznaczeniach.

Dla tak określonego stanu koła (wzory (56a) i (56b)) formułujemy rozwinięty opis przyspieszenia koła i reakcji więzów według wzorów (25):

$$m\ddot{x} = -f'(x)\lambda - \mu \left\|G\right\| \lambda\tau \tag{57}$$

$$m\ddot{y} = -mg + \lambda - \mu \left\|G\right\| f'(x)\lambda\tau \tag{58}$$

$$J\ddot{\varphi} = -\mu \left\|G\right\| \lambda r\tau \tag{59}$$

$$\lambda = \left[\lambda - \rho a_n\right]^+, \quad \rho > 0 \tag{60}$$

$$\tau = \begin{cases} \operatorname{sign} v_t & \text{gdy} \quad v_t \neq 0 \\ \Pi\left(\tau + \rho a_t\right), \ \rho > 0 & \text{gdy} \quad v_t = 0 \end{cases} \tag{61}$$

gdzie wprowadzono następujące oznaczenia:

$$G(x) := \left[-f'(x) \quad 1 \quad 0\right]^{\mathrm{T}} \text{ (wg wzoru (26c))} \tag{62}$$

$\left\|G\right\|\lambda$ – siła nacisku koła na krawężnik

v_t – styczna składowa prędkości punktu koła stykającego się z krawężnikiem

$$v_t := \dot{x} + f'(x)\dot{y} + r\dot{\varphi} \tag{63}$$

a_n – normalna składowa przyspieszenia punktu koła stykającego się z krawężnikiem

$$a_n := \ddot{y} - f'(x)\ddot{x} + f''(x)\dot{x}^2 \tag{64a}$$

z warunku więzów (11) wynika, że wartość tego przyspieszenia może być większa bądź równa zero. Jeżeli we wzorze (64a) uwzględnimy równania (57) i (58), to otrzymamy

$$a_n := \frac{1}{m}\left(\left\|G\right\|^2 \lambda + (\gamma - g)\right), \ \gamma := f''(x)\dot{x}^2 \tag{64b}$$

gdzie a_t – styczna składowa przyspieszenia punktu koła stykającego się z krawężnikiem:

$$a_t := \ddot{x} + f'(x)\ddot{y} + r\ddot{\varphi} + f''(x)\dot{x}\dot{y} \tag{65a}$$

Jeżeli w powyższym równaniu uwzględnimy równania (57), (58) i (59) oraz wykonamy stosowne przekształcenia, to uzyskamy:

$$a_t := -\left(\frac{1}{m}\|G\|^2 + \frac{r^2}{J}\right)\mu\,\|G\|\lambda\tau + f'(x)g + f''(x)\dot{x}\dot{y} \tag{65b}$$

Aby wyznaczyć wartość mnożnika λ, należy do równania (60) wstawić wyrażenie a_n według wzoru (64b); stąd można bezpośrednio obliczyć wartość λ.

Gdy wartość prędkości poślizgu koła po krawężniku v_t (według wzoru (63)) jest różna od zera, to ze wzoru (61_1) otrzymujemy wartość mnożnika $\tau = \mathrm{sing} v_t$. Jeżeli $v_t = 0$, to do równania (61_2) należy wstawić wyrażenie a_t według wzoru (65b); stąd wynika wartość mnożnika τ.

Po wyznaczeniu wartości mnożników λ i τ obliczamy wartości przyspieszeń $\ddot{x}$, $\ddot{y}$ i $\ddot{\varphi}$ według wzorów (57), (58) i (59).

Zderzenie koła z krawężnikiem powstaje, gdy prędkość v_n według wzoru (56b) jest ujemna, czyli $v_n < 0$ (patrz rys. 4a). Efekt tego zjawiska, przy założeniu hipotezy zderzenia plastycznego, opisują równania:

$$m\dot{x}^+ - m\dot{x}^- = -f'(x)\tilde{\lambda} - \mu\,\|G\|\tilde{\lambda}\tau \tag{66}$$

$$m\dot{y}^+ - m\dot{y}^- = \tilde{\lambda} - \mu\, f'(x)\|G\|\tilde{\lambda}\tau \tag{67}$$

$$J\dot{\varphi}^+ - J\dot{\varphi}^- = -\mu\,\|G\|\tilde{\lambda} r\tau \tag{68}$$

$$\tilde{\lambda} = \left[\tilde{\lambda} - \rho\, v_n^+\right]^+, \quad \rho > 0 \tag{69}$$

$$\tau = \Pi\left(\tau + \rho\, v_t^+\right), \ \rho > 0 \tag{70}$$

Rozwiązanie tych równań względem $\dot{x}^+$, $\dot{y}^+$, $\dot{\varphi}^+$, $\tilde{\lambda}$, τ wyznacza prędkość koła po zderzeniu plastycznym oraz impulsy siły reakcji. W celu wyznaczenia rozwiązania ustalamy wyrażenia określające v_n^+ i v_t^+ w zależności od mnożników $\tilde{\lambda}$ i τ:

$$v_n^+ := \dot{y}^+ - f'(x)\dot{x}^+ = v_n^- + \frac{1}{m}\|G\|^2\,\tilde{\lambda} \tag{71}$$

$$v_t^+ := \dot{x}^+ - f'(x)\dot{y}^+ + r\dot{\varphi}^+ = v_t^- + \left(\frac{1}{m}\|G\|^2 + \frac{r^2}{J}\right)\mu\,\|G\|\tilde{\lambda}\tau \tag{72}$$

Po uwzględnieniu wzoru (71) w równaniu (69) otrzymujemy wartość mnożnika $\tilde{\lambda}$. Następnie z równania (70) oraz z zależności (72) wyznaczamy wartości τ oraz v_t^+. Po wyznaczeniu mnożników obliczamy wartości poszukiwanych prędkości po zderzeniu plastycznym $\dot{x}^+$, $\dot{y}^+$ i $\dot{\varphi}^+$.

Jeżeli rozpatrujemy zderzenie sprężyste albo sprężysto-plastyczne, to należy rozważyć opis drugiej fazy zderzenia (35), analogiczny do wzorów (50)÷(55):

$$\tilde{\lambda}^* = \beta\,\tilde{\lambda}, \ \beta \in (0,1] \tag{73}$$

$$m\dot{x}^* = m\dot{x}^+ - f'(x)\tilde{\lambda}^* - \mu \|G\| \tilde{\lambda}^* \tau^* \tag{74}$$

$$m\dot{y}^* = m\dot{y}^+ - \mu f'(x) \|G\| \tilde{\lambda}^* \tau^* \tag{75}$$

$$J\dot{\varphi}^* = J\dot{\varphi}^+ - \mu \|G\| \tilde{\lambda}^* r\tau^* \tag{76}$$

$$\tau^* = \Pi\left(\tau^* + \rho v_t^*\right), \quad \rho > 0 \tag{77}$$

$$v_t^* := v_t^+ + \left(\frac{1}{m}\|G\|^2 + \frac{r^2}{J}\right) \mu \|G\| \tilde{\lambda}^* \tau^* \tag{78}$$

Na podstawie równań (77) i (78) można bezpośrednio wyznaczyć wartość mnożnika τ^*, a następnie wartości prędkości koła.

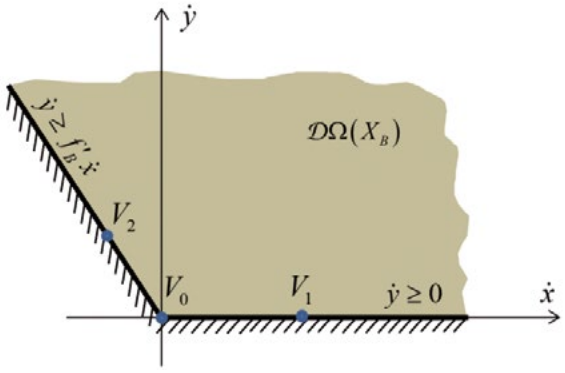
Jeśli koło zajmuje położenie osobliwe, określone punktem B (rys. 8), to opis oddziaływania na koło komplikuje się, gdyż styka się ono z więzami w dwóch punktach B_1 i B_2 zaznaczonych na rysunku 8. Położenie koła wyznaczają współrzędne środka $x = l_1 + a$, $y = r$, natomiast opis zbioru dopuszczalnych prędkości koła w tym położeniu jest następujący:

$$\mathrm{D}\Omega\left(X_B\right) := \left\{\dot{x}, \dot{y} \in R^1 : \quad -f'_B \dot{x} + \dot{y} \geq 0,\ \dot{y} \geq 0\right\} \tag{79}$$

gdzie f'_B – wartość pochodnej funkcji f dla $x = l_1 + a$

$$f'_B := -\frac{a}{\sqrt{r^2 - a^2}}, \quad a < r \tag{80}$$

Na rysunku 9 zamieszczono schematyczny wykres zbioru $\mathrm{D}\Omega$.



RYS. 9. **Wykres zbioru DΩ w osobliwym punkcie B (por. rys. 8)**
FIG. 9. **The graph of the set DΩ at a singular point B (see Fig. 8)**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Na podstawie analizy rysunku 9 widać, że w rozpatrywanym położeniu więzy działają, czyli ograniczają przyspieszenia tylko wtedy, gdy wektor prędkości należy do brzegu zbioru $\mathrm{D}\Omega$, co zapisano we wzorze (8a). Na rysunku 9 zaznaczono trzy przykładowe wektory prędkości $V := \left[\dot{x}, \dot{y}\right]^{\mathrm{T}}$ leżące na brzegu. W przypadku wektorów V_1 i V_2 działa tylko jedno ograniczenie, natomiast dla wektora V_0 – oba.

Działanie pojedynczych ograniczeń opisano wcześniej, a teraz zajmiemy się sytuacją określoną prędkością V_0, czyli gdy $\dot{y} - f_B' \dot{x} = 0$ oraz $\dot{y} = 0$. Z tych warunków wynika, że postępowa prędkość jest równa zero, ale zakładamy, że koło się obraca, czyli $\dot{\varphi} \neq 0$. W takiej sytuacji koło ślizga się po ograniczeniach.

W opisanym stanie przyspieszenie koła jest ograniczane, a zbiór dopuszczalnych przyspieszeń ma postać

$$\mathrm{D}^2\Omega\left(X_B, V_B\right) := \left\{\ddot{x}, \ddot{y} \in R^1 : \quad -f_B' \ddot{x} + \ddot{y} \geq 0,\ \ddot{y} \geq 0\right\} \tag{81}$$

Równania określające przyspieszenie i siły reakcji mają natomiast teraz postać

$$m\ddot{x} = -f_B' \lambda_1 - \mu \left\|G\right\| \lambda_1 \tau_1 - \mu\, \lambda_2 \tau_2 \tag{82}$$

$$m\ddot{y} = -mg + \lambda_1 - \mu \left\|G\right\| f_B' \lambda_1 \tau_1 + \lambda_2 \tag{83}$$

$$J\ddot{\varphi} = -\mu \left\|G\right\| r \lambda_1 \tau_1 - \mu\, r \lambda_2 \tau_2 \tag{84}$$

$$\lambda_1 = \left[\lambda_1 - \rho\, a_1^n\right]^+, \quad \rho > 0 \tag{85}$$

$$\lambda_2 = \left[\lambda_2 - \rho\, a_2^n\right]^+, \quad \rho > 0 \tag{86}$$

gdzie przez λ_i, τ_i, dla $i \in \{1,2\}$, oznaczono mnożniki reakcji działające na koło w punktach B_i pokazanych na rysunku 8; ponadto a_1^n, a_2^n – składowe normalne przyspieszenia koła punktach B_1 i B_2:

$$a_1^n := \ddot{y} - f_B' \ddot{x}, \ a_2^n := \ddot{y} \tag{87}$$

W związku z tym, że przyjęto, iż $\dot{\varphi} \neq 0$, to

$$\tau_1 = \tau_2 = \operatorname{sign} \dot{\varphi} \tag{88}$$

Na podstawie przytoczonych powyżej równań należy wyznaczyć wartości $\ddot{x}$, $\ddot{y}$, $\ddot{\varphi}$ oraz λ_1 i λ_2. W tym celu ze wzorów (87) oraz równań (82), (83) i (84) ustalamy wyrażenia:

$$a_1^n = -g + \frac{1}{m}\left[\left\|G\right\|^2 \lambda_1 + \left(\mu\, f_B' \tau_2 + 1\right)\lambda_2\right] \tag{89}$$

$$a_2^n = -g + \frac{1}{m}\left[\left(1 - \mu\, f_B' \left\|G\right\| \tau_1\right)\lambda_1 + \lambda_2\right] \tag{90}$$

Wstawiając wyrażenie (89) do wzoru (85) oraz przyjmując $\rho := m/\|G\|^2$, otrzymujemy:

$$\lambda_1 = \|G\|^{-2}\left[mg - \gamma_2\lambda_2\right]^+, \text{ jeśli } \gamma_2 := \mu f_B'\tau_2 + 1 \tag{91}$$

Podobnie ze wzorów (90) i (86), gdy $\rho := m$, otrzymujemy:

$$\lambda_2 = \left[mg - \gamma_1\lambda_1\right]^+, \text{ jeśli } \gamma_1 := 1 - \mu f_B'\|G\|\tau_1 \tag{92}$$

Rozwiązanie równań (91) i (92) wyznacza wartości mnożników λ_1 i λ_2, natomiast wartości przyspieszeń $\ddot{x}$, $\ddot{y}$, $\ddot{\varphi}$ obliczamy ze wzorów (82), (83) i (84), po uwzględnieniu (88).

Z analizy równań (91) i (92) wynika, że jeżeli:

$$\dot{\varphi} < 0, \text{ czyli } \tau_1 = \tau_2 = -1, \text{ to} \tag{92a}$$

$$\lambda_1 = 0, \lambda_2 = mg, \tag{92b}$$

czyli otrzymuje się oczywisty rezultat: koło odjeżdża od krawężnika, czyli $\ddot{x} = -\mu g$, $\ddot{y} = 0$, $\ddot{\varphi} = -\frac{1}{J}\mu mgr$.

Jeżeli

$$\dot{\varphi} > 0, \text{ czyli } \tau_1 = \tau_2 = 1 \tag{93a}$$

to są możliwe dwa rozwiązania:

$$\lambda_1 > 0, \lambda_2 > 0, \text{ gdy } \|G\|^2 > \gamma_1 \tag{93b}$$

$$\lambda_1 = \frac{mg}{\|G\|^2}, \ \lambda_2 = 0, \text{ gdy } \|G\|^2 \le \gamma_1 \tag{93c}$$

W przypadku określonym wzorem (93b) środek koła nie przemieszcza się – $\ddot{x} = \ddot{y} = 0$, ale obraca – $\dot{\varphi} > 0$, $\ddot{\varphi} < 0$ (buksuje); natomiast w drugim przypadku (93c) koło wspina się na krawężnik, gdyż $\ddot{x} < 0$, $\ddot{y} > 0$.

W rozpatrywanym położeniu koła powstaje zderzenie z krawężnikiem, gdy prędkości $\dot{x}^-$ oraz $\dot{y}^-$ nie spełniają warunków wymienionych we wzorze (79), czyli nie należą do zbioru $\mathsf{D}\ \Omega(X_B)$, którego wykres pokazano na rysunku 9. Powstające wtedy zderzenie plastyczne opisują następujące relacje, które zamieszczono w rozdziale 5, we wzorach (32)÷(34):

$$m\dot{x}^+ = m\dot{x}^- - f_B'\tilde{\lambda}_1 - \mu\|G\|\tilde{\lambda}_1\tau_1 - \mu\,\tilde{\lambda}_2\tau_2 \tag{94}$$

$$m\dot{y}^+ = m\dot{y}^- + \tilde{\lambda}_1 - \mu f_B'\|G\|\tilde{\lambda}_1\tau_1 + \tilde{\lambda}_2 \tag{95}$$

$$J\dot{\varphi}^+ = J\dot{\varphi}^- - \mu\|G\|r\tilde{\lambda}_1\tau_1 - \mu\, r\tilde{\lambda}_2\tau_2 \tag{96}$$

$$\tilde{\lambda}_1 = \left[\tilde{\lambda}_1 - \rho v_{n1}^+\right]^+, \quad \rho > 0 \tag{97}$$

$$\tilde{\lambda}_2 = \left[\tilde{\lambda}_2 - \rho v_{n2}^+\right]^+, \quad \rho > 0 \tag{98}$$

$$\tau_1 = \Pi\left(\tau_1 + \rho v_{t1}^+\right), \ \rho > 0 \tag{99}$$

$$\tau_2 = \Pi\left(\tau_2 + \rho v_{t2}^+\right), \ \rho > 0 \tag{100}$$

gdzie wprowadzono oznaczenia:

$$v_{n1}^- := \dot{x}^- - f_B' \dot{y}^-, \ v_{n1}^+ := \dot{x}^+ - f_B' \dot{y}^+ \tag{101}$$

$$v_{n2}^- := \dot{y}^-, \ v_{n2}^+ := \dot{y}^+ \tag{102}$$

$$v_{t1}^- := \dot{x}^- + f_B' \dot{y}^- + r\dot{\varphi}^-, \ v_{t1}^+ := \dot{x}^+ + f_B' \dot{y}^+ + r\dot{\varphi}^+ \tag{103}$$

$$v_{t2}^- := \dot{x}^- + r\dot{\varphi}^-, \ v_{t2}^+ := \dot{x}^+ + r\dot{\varphi}^+ \tag{104}$$

Z przytoczonych wyżej równań należy wyznaczyć prędkości kół po zderzeniu $\dot{x}^+$, $\dot{y}^+$, $\dot{\varphi}^+$ oraz mnożniki impulsów reakcji $\tilde{\lambda}_1$, $\tilde{\lambda}_2$, τ_1, τ_2. Ponieważ jest to stosunkowo złożony układ nieliniowych równań, to do wyznaczenia jego rozwiązania wykorzystano metodę iteracyjną. W tej metodzie poszukuje się wektora $Y := \left[\dot{x}^+, \dot{y}^+, \dot{\varphi}^+, \tilde{\lambda}_1, \tilde{\lambda}_2, \tau_1, \tau_2\right]^{\mathrm{T}} \in R^7$, który spełnia równanie:

$$Y = \Psi(Y) \tag{105}$$

gdzie $\Psi : R^7 \to R^7$ oznacza odwzorowanie określone prawymi stronami podanych wyżej równań.

Gdy rozpatrujemy zderzenie sprężyste lub sprężysto-plastyczne, to na podstawie równań (35) należy sformułować stosowne relacje uwzględniające szczegółowe dane określające stan koła w punkcie *B*. Na tej podstawie ustalamy następujące równania:

$$\tilde{\lambda}_i^* = \beta\, \tilde{\lambda}_i, \ i = 1, 2 \tag{106}$$

$$m\dot{x}^* = m\dot{x}^+ - f_B' \tilde{\lambda}_1^* - \mu \left\|G_1\right\| \tilde{\lambda}_1^* \tau_1^* - \mu\, \tilde{\lambda}_2^* \tau_2^* \tag{107}$$

$$m\dot{y}^* = m\dot{y}^+ + \tilde{\lambda}_1^* - \mu\, f_B' \left\|G_1\right\| \tilde{\lambda}_1^* \tau_1^* + \tilde{\lambda}_2^* \tag{108}$$

$$J\dot{\varphi}^* = J\dot{\varphi}^+ - \mu \left\|G_1\right\| r\, \tilde{\lambda}_1^* \tau_1^* - \mu\, \tilde{\lambda}_2^* \tau_2^* \tag{109}$$

$$\tau_i^* = \Pi\left(\tau_i^* + \rho v_{ti}^*\right), \ \rho > 0, \ i = 1, 2 \tag{110}$$

$$v_{t1}^{*} := \dot{x}^{*} + f_B' \dot{y}^{+} + r\dot{\varphi}^{*} \tag{111}$$

$$v_{t2}^{*} := \dot{x}^{*} + r\dot{\varphi}^{*} \tag{112}$$

Na podstawie powyższych relacji można uzyskać dwa nieliniowe równania względem τ_1^* oraz τ_2^*, które rozwiązujemy metodą iteracyjną. Po ustaleniu ich wartości można wyznaczyć wartości prędkości po drugiej fazie rozpatrywanego zderzenia $\dot{x}^+$, $\dot{y}^+$, $\dot{\varphi}^+$.

Relacje podane w niniejszym podrozdziale dotyczą wyznaczenia przyspieszenia koła oraz jego prędkości po zderzeniu wtedy, gdy koło znajduje się w segmentach IV, II i III (rys. 7) oraz w punkcie *B* (rys. 8). Nie podajemy równań dla segmentu I, gdyż są one analogiczne do równań segmentu IV; natomiast równania w osobliwym punkcie *A* są podobne do opisanych wyżej równań dla punktu *B*.

Wyniki obliczeń komputerowych

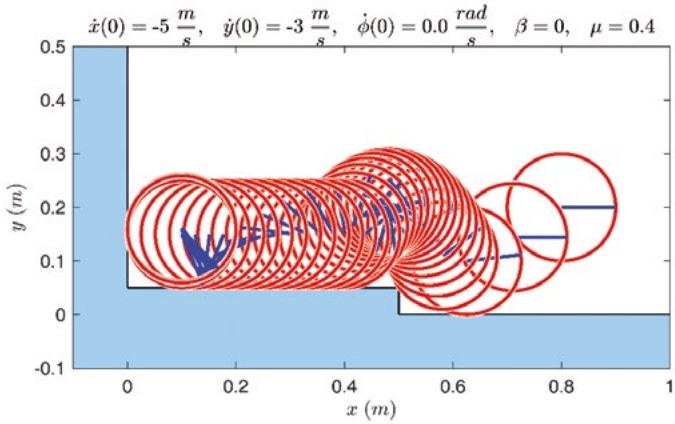
Rozpatrujemy płaski ruch nieodkształcalnego koła, na które działają ograniczenia przedstawione na rysunkach 1, 2 i 7. Symulację ruchu koła wykonujemy, przyjmując następujące wartości parametrów koła: masa $m = 10$ kg, moment bezwładności $J = 0{,}05\ \text{kg m}^2$ i promień $r = 0{,}1$ m. Wymiary zaznaczone na rysunku 1 wynoszą $h = 5$ cm oraz $l = 0{,}50$ m. Poza tym przyjmujemy, że przyspieszenie ziemskie wynosi $g = 9{,}81\ \text{m/s}^2$, a warunki początkowe są określone dwoma wektorami

$$X_o = \begin{bmatrix} 0{,}8\ \text{m} \\ 0{,}2\ \text{m} \\ 0\ \text{rad} \end{bmatrix},\ V_o = \begin{bmatrix} -5\ \text{m/s} \\ -3\ \text{m/s} \\ 0\ \text{rad/s} \end{bmatrix} \tag{113}$$

Na podstawie relacji zamieszczonych w rozdziale 4 ustalono dla każdego segmentu brzegu ograniczeń (rys. 7) zestawy relacji określających przyspieszenie koła i siły reakcji oraz relacje określające nieciągłe zmiany prędkości koła spowodowane zderzeniem.

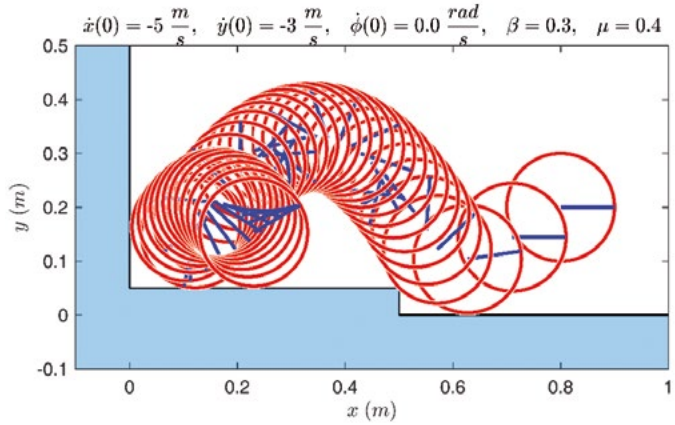
Symulację ruchu koła przeprowadzono w przedziale czasu $t \in (0,1]$ s. Przyjęto wartość współczynnika tarcia pomiędzy kołem i brzegiem więzów $\mu = 0{,}4$. Rozpatrywano dwa zadania, które różniły się jedynie wartością współczynnika restytucji β. W pierwszym zadaniu symulowano ruch koła przy założeniu hipotezy zderzeń plastycznych $\beta = 0$. W przypadku drugiego zadania przyjęto $\beta = 0{,}3$, co odpowiada hipotezie zderzenia sprężysto-plastycznego.

Sekwencje położeń koła w trakcie symulacji obrazują rysunek 10 (zderzenia plastyczne) i rysunek 11 (zderzenia sprężysto-plastyczne). Na tych rysunkach podano również warunki początkowe koła; wynika stąd, że w pierwszym okresie ruchu koło porusza się ruchem postępowym, a po pierwszym zderzeniu zaczyna się obracać.



RYS. 10. Sekwencja położeń koła w trakcie symulacji (zderzenia plastyczne)
FIG. 10. The sequence of wheel positions during computer simulation (plastic collisions)

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

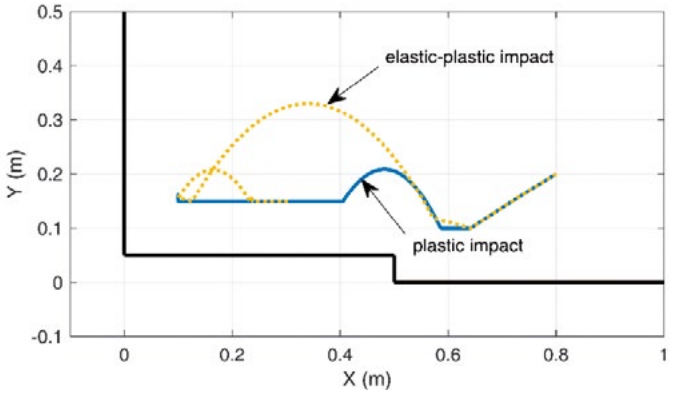


RYS. 11. Sekwencja położeń koła w trakcie symulacji (zderzenia sprężysto-plastyczne)
FIG. 11. The sequence of wheel positions during computer simulation (resilient-plastic collisions)

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Na kolejnym rysunku (12) przedstawiono dwie trajektorie (tory ruchu) zakreślone przez środek koła w przypadku dwóch analizowanych zadań.

Dwa ostatnie wykresy obrazują przebiegi czasowe położeń środka koła oraz kąta obrotu (rys. 13) a także przebiegi czasowe prędkości zmiany tych wielkości (rys. 14). Wyniki zilustrowane na zamieszczonych rysunkach wskazują na istotne różnice w zachowaniu się koła w zależności od przyjętej hipotezy zderzenia.

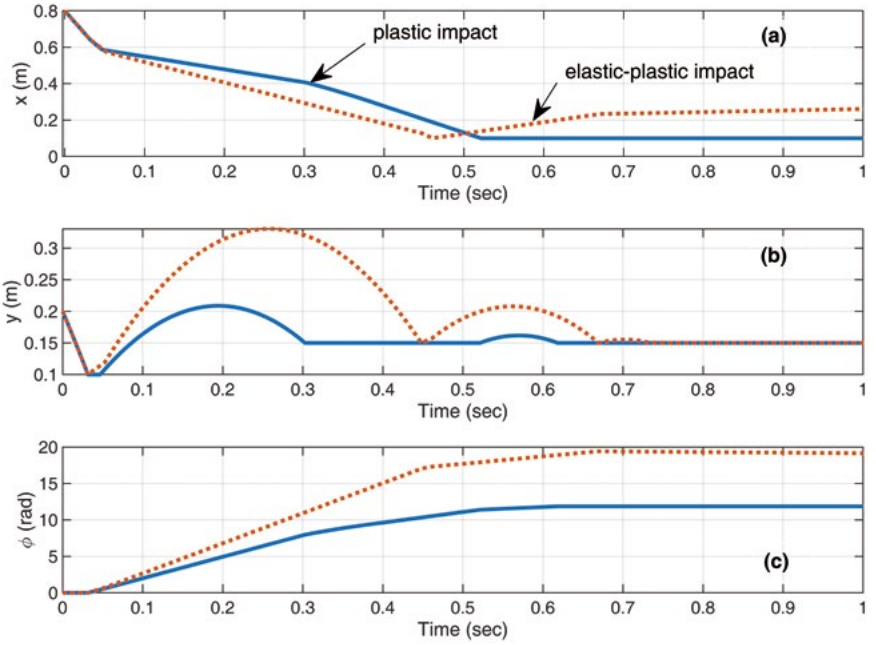


RYS. 12. Trajektorie zakreślone przez środek koła: zderzenia plastyczne (linia ciągła), zderzenia sprężysto-plastyczne (linia przerywana)

FIG. 12. Trajectories circled by the center of the circle: plastic collisions (solid line), resilient-plastic collisions (dashed line)

ŹRÓDŁO: opracowanie własne.

SOURCE: own elaboration.

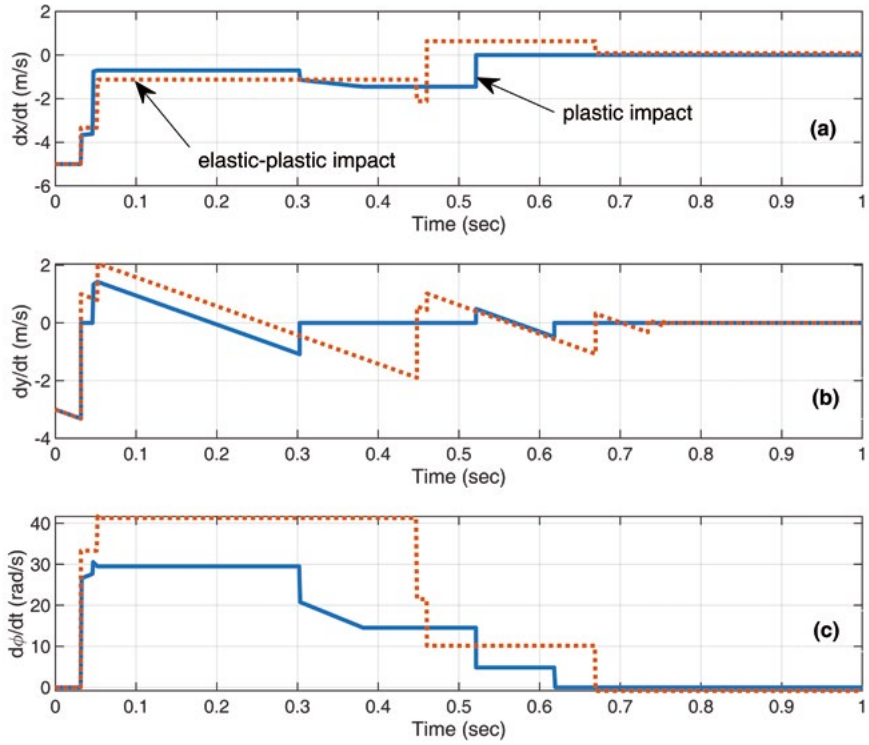


RYS. 13. Przebiegi czasowe położeń środka koła (a) i (b) oraz kąta obrotu (c): zderzenia plastyczne (linia ciągła), zderzenia sprężysto-plastyczne (linia przerywana)

FIG. 13. The time responses of wheel position (a) and (b) and angle of rotation (c): plastic collisions (solid line), resilient-plastic collisions (doshed line)

ŹRÓDŁO: opracowanie własne.

SOURCE: own elaboration.



RYS. 14. Przebiegi czasowe prędkości środka koła (a) i (b) oraz prędkości kątowej (c): zderzenia plastyczne (linia ciągła), zderzenia sprężysto-plastyczne (linia przerywana)

FIG. 14. The time responses of wheel center velocity (a) and (b) and angular velocity (c): plastic collisions (solid line), resilient-plastic collisions (doshed line)

ŹRÓDŁO: opracowanie własne.

SOURCE: own elaboration.

7. Wnioski

W pracy przedstawiono analizę niegładkiego zagadnienia ruchu układu mechanicznego na przykładzie koła, którego ruch jest ograniczany.

Pokazano, że sformułowanie niegładkiego zagadnienia ruchu składa się z relacji wyznaczających przyspieszenia ciał oraz z relacji określających nieciągłe zmiany prędkości ciał spowodowane zderzeniem. Opisy tych relacji ustalono według hipotez określających oddziaływania (reakcje) między stykającymi się ciałami nieodkształcalnymi.

W rozpatrywanym tu przykładzie z kołem przyjęte hipotezy mają jasną interpretację, dzięki czemu sformułowanie niegładkiego zagadnienia ruchu jest stosunkowo proste. W przypadku ciała sztywnego w przestrzeni trójwymiarowej opis reakcji jest bardziej złożony, a poza tym w szczególnych położeniach ciała nie można jednoznacznie wyznaczyć jego przyspieszenia i siły reakcji. Takie osobliwe stany układu wynikają z przybliżonego charakteru hipotez określających siły działające między stykającymi się ciałami, a zwłaszcza dotyczy to siły tarcia.

Niegładkie zagadnienia ruchu poszerzają klasyczną problematykę mechaniki teoretycznej. Dzięki temu zwiększa się zakres metod modelowania i symulacyjnych badań ruchu oraz obciążeń maszyn, pojazdów i elementów konstrukcji inżynierskich.

Literatura

1. Grzesikiewicz W, Zbiciak A. *Opis ruchu układu mechanicznego z więzami jednostronnymi*. Warszawa: Oficyna Wydawnicza Politechniki Warszawskiej; 2018.
2. Painlevé P. *Leçons sur le frottement*. Paris: Hermann; 1895.
3. Routh EJ. *Dynamics of a system of rigid bodies*. London: MacMillan; 1905.
4. Jean M, Moreau JJ. *Unilaterality and dry friction in the dynamics of rigid body collections*. Proc. Contact Mechanics International Symposium. Lausanne: Presses Polytechniques et Universitaires Romandes; 1992 ; 31-48.
5. Jean M. The nonsmooth contact dynamics method. *Computer Methods in Applied Mechanics and Engineering*. 1999; 177: 235-257. Special issue on computational modeling of contact and friction, J.A.C. Martins and A. Klarbring, editors.
6. Moreau JJ, Panagiotopoulos PD. Eds.: Non-Smooth Mechanics and Applications. CISM Courses and Lectures, vol. 302.Wien: Springer; 1988.
7. Panagiotopoulos PD. *Inequality Problems in Mechanics and Applications: Convex and Nonconvex Energy Functions*. Basel: Birkhäuser; 1985.
8. Panagiotopoulos PD. *Hemivariational Inequalities: Applications in Mechanics and Engineering*. Springer-Verlag; 1993.
9. Grzesikiewicz W. Dynamika układów mechanicznych z więzami. *Prace Naukowe Politechniki Warszawskiej*, Mechanika. 1990; 117.
10. Grzesikiewicz W, Wakulicz A. Dynamics of non-smooth mechanical system. *Machine Dynamics Problems*. 1999; 23(1): 25-37.
11. Grzesikiewicz W, Wakulicz A, Zbiciak A. *Succession of constraint imposed on time function. Monograph: Polioptimization and Computer Aided Design*, vol. 7. Koszalin: Publishing House of Koszalin University of Technology; 2009; 49-56.
12. Stronge WJ. Smooth dynamics of oblique impact with friction. *International Journal of Impact Engineering*. 2013; 51: 36-49.
13. Stewart DE. Rigid-Body Dynamics with Friction and Impact. *SIAM Review, Society for Industrial and Applied Mathematics*. 2000; 42(1): 3-39.

14. Le xuan Anh: *Dynamics of Mechanical Systems with Coulomb Friction*. Berlin –Heidelberg: Springer-Verlag; 2003.
15. Rutczyńska-Wdowiak K. *Replacement strategies of genetic algorithm in parametric identification of induction motor.* 22nd International Conference on Methods and Models in Automation and Robotics; 2017; 971-975.
16. Rutczyńska-Wdowiak K. The generating new individuals of the population in the parametric identification of the induction motor problem with the use of the genetic algorithm. *Technical Transactions*. 2019; 2: 5-13.

Streszczenie

W pracy przedstawiono problem modelowania i symulacji niegładkiego zagadnienia ruchu. Rozpatrujemy ruch układu, na który działają niedoskonałe więzy jednostronne. Matematyczny opis ruchu takiego układu ma postać niegładkiego zagadnienia początkowego (Cauchy'ego). Niegładkość tego zagadnienia oznacza, że jego rozwiązanie wyznacza funkcja absolutnie ciągła, czyli mająca nieciągłą pierwszą pochodną. Z tego powodu, obok równań ruchu określających przyspieszenie i siłę reakcji, sformułowano dodatkowe zagadnienie zderzenia, opisujące skokową zmianę prędkości oraz impuls reakcji. Do wyznaczenia przybliżonego rozwiązania sformułowanego zagadnienia ruchu koła opracowano oryginalną metodę numeryczną oraz program do obliczeń komputerowych służący do symulacji ruchu koła. Przedstawiono wybrane wyniki ilustrujące przebieg przemieszczeń i prędkości oraz sił reakcji więzów.

Słowa kluczowe: mechanika niegładka, więzy nieidealne, zderzenie, tarcie, dynamika bryły sztywnej

Summary
The modeling and simulation of non-smooth motion problem

The paper presents the problem of modeling and simulation of non-smooth motion issue. We consider the motion of the system into which imperfect unilateral constraints operate. The mathematical description of the movement of such a system takes the form of a non-smooth initial problem (Cauchy's). The non-smoothness of this problem means that its solution is determined by an absolutely continuous function, i.e. having a discontinuous first derivative. For this reason, along with motion equations determining acceleration and reaction force, an additional collision issue was formulated, describing the step change in speed and the impulse of reaction. To determine the approximate solution of the formulated issue of the wheel motion, the original numerical method was developed as well as a computer calculation program for simulating wheel motion. Selected results illustrating the displacements and velocities as well as reaction forces of constraints are presented.

Keywords: non-smooth mechanics, non-ideal constraints, impact, friction, rigid-body dynamics

DOI: 10.24427/978-83-66391-28-4_2

Stochastyczny model ogólnego cyklu życia ataku cybernetycznego

Romuald HOFFMANN[*]

1. Wprowadzenie

Bez wątpienia dzisiejszy rozwój informatyki jest jednym z najważniejszych czynników rozwoju współczesnego świata. Oddziałuje on na wiele procesów społecznych, gospodarczych i militarnych. Wpływ ten należy postrzegać zarówno w sensie pozytywnym, jak i negatywnym. Do niepożądanych efektów jego oddziaływania należy zaliczyć rosnącą cyberprzestępczość. Niestety do grona indywidualnych i zorganizowanych cyberprzestępców dołączyły państwa, dla których ataki w cyberprzestrzeni stały się elementem prowadzonej agresywnej polityki gospodarczej i militarnej, wywołując w ten sposób reakcje obronne pozostałych. Właściwie już teraz można dostrzec swoisty wyścig zbrojeń w cyberprzestrzeni, który zdaniem autora w niedalekiej przyszłości stanie się normą, prowadząc w konsekwencji do poważnych konfliktów militarnych. Nie pomylimy się, twierdząc, że rywalizacja państw w cyberprzestrzeni stała się faktem i zjawisko to obecnie bardzo szybko narasta. Analizując dotychczasowe przypadki ataków cybernetycznych, dostrzega się w nich pewną systematykę – zostały one przeprowadzone według pewnego schematu, który śmiało można określić mianem procesu cyberataku. Pomimo tego, jak często mówi się i pisze o cyberatakach, wiele organizacji odbiera atak cybernetyczny jako krótkotrwałe zdarzenie, któremu prawie nie da się przeciwstawić. W rzeczywistości atak cybernetyczny nie jest jednak aktem chwilowym, ale procesem, na który składa się zbiór czynności, które należy wykonać w odpowiedniej kolejności i które mają swoje czas i miejsce trwania [1] [10]. W zależności od celu ataku czynności te łączy się w logiczne grupy i realizuje etapowo, tworząc w ten sposób proces ataku cybernetycznego. Proces ten ma skończony czas trwania i nazywamy go cyklem życia ataku cybernetycznego (ang. *cyber attack life cycle*, *cyber kill chain*). Znajomość tego cyklu może umożliwić np. oszacowanie prawdopodobieństwa ataku, średniego czasu trwania ataku lub średniego czasu do kompromitacji systemu informatycznego (ang. *time-to-compromise*), a ostatecznie kosztów ataku. Znając te wymienione i inne charakterystyki procesu, możemy starać się odpowiedzieć na kluczowe pytania, np. o to, kiedy nastąpi prawdopodobny atak, jakiej fazie ataku podlegamy i z jakim prawdopodobieństwem.

[*] Wojskowa Akademia Techniczna

W pracy przedstawiono przywołany z [1] opis ogólnego cyklu życia ataku cybernetycznego składającego się z następujących siedmiu faz: identyfikacja i definicja potrzeb – planowanie wstępne, rozpoznanie, uzbrojenie, dostarczenie, uruchomienie i kontrola kodu złośliwego, realizacja celów, zakończenie ataku i zatarcie śladów. Wspomniany cykl modelowano w [1] z wykorzystaniem łańcuchów Markowa z ciągłym parametrem czasu. W modelu źródłowym przyjęto założenie wykładniczego rozkładu prawdopodobieństwa czasów trwania poszczególnych faz cyklu cyberataku, który może stanowić ograniczenie wykorzystania modelu w praktyce. Wobec tego w pracy zaproponowano stochastyczny model ogólnego cyklu życia ataku cybernetycznego, zakładając, że czasy trwania poszczególnych faz ataku mają dowolny ciągły rozkład prawdopodobieństwa. Ponadto w proponowanym modelu uwzględniono powtarzalność ataków cybernetycznych przejawiającą się w powtarzalności cyklu ataku cybernetycznego.

2. Ogólny cykl życia ataku cybernetycznego

W pracy [1] zaproponowano ogólny cykl życia ataku cybernetycznego, składający się z następujących faz: (S_1) identyfikacja i definicja celów ataku – planowanie wstępne, (S_2) rozpoznanie (ang. *reconnaissance*), (S_3) uzbrojenie (ang. *weaponization*), (S_4) dostarczenie kodu złośliwego (ang. *delivery*), (S_5) uruchomienie i kontrola kodu złośliwego (ang. *cyber execution and command & control*), (S_6) realizacja celów (ang. *action, achieve objectives*) oraz (S_7) zakończenie ataku i zatarcie śladów. Propnowany w [1] cykl życia różni się od dotychczas opisywanych w literaturze [2, 3, 4, 5, 6] dwoma dodatkowymi fazami: S_1 i S_2. Te dwie fazy uwzględniają występujące w praktyce czynności planowania i zakończenia ataku. Pozostałe fazy cyklu mają swoje odpowiedniki w literaturze, stąd też podano w nawiasach ich angielskie nazwy. Tabela 1 zawiera opis faz ogólnego cyklu życia ataku cybernetycznego [1].

TAB. 1. Fazy ogólnego cyklu życia ataku cybernetycznego [1]
TAB. 1. Phases of general cyber-attack life cycle [1]

Nazwa fazy	Opis fazy ataku cybernetycznego
Identyfikacja i definicja celów – planowanie wstępne (S_1)	Identyfikacja i określenie potrzeb agresora/atakującego, np. biznesowych, politycznych itp. Faza ta powinna wystąpić nawet wtedy, gdyby był to tylko pomysł przestępcy na przejęcie np. konta ofiary na Twitterze. Na pewno występuje wówczas, gdy np. grupa przestępcza lub jakaś organizacja planuje swoje działania, wynika z przyjętej szerszej strategii państwa dotyczącej działań w cyberprzestrzeni itp.

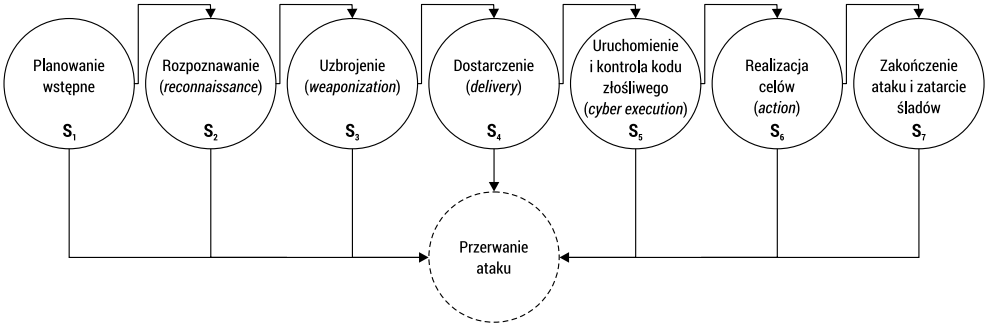
Nazwa fazy	Opis fazy ataku cybernetycznego
Rozpoznanie (ang. *reconnaissance*) (S_2)	Identyfikacja i dobór celów ataków (technicznych) poprzez rozpoznanie docelowego środowiska, np. skanowanie portów TCP, indeksowanie witryn internetowych, materiałów konferencyjnych, list adresów e-mail, sieci społecznościowych, informacji na temat stosowanych (specyficznych) technologii, socjotechniczne wyłudzenie informacji i danych itp.
Uzbrojenie (ang. *weaponization*) (S_3)	Przygotowanie cyberbroni, tzn. specjalnego oprogramowania, np. zintegrowanie koni trojańskich z innym złośliwym kodem (ang. *exploit*) w celu stworzenia możliwego do dostarczenia ładunku za pomocą automatycznego narzędzia (ang. *weaponizer*). W przypadku, gdy nie zachodzi potrzeba budowy lub skonfigurowania pakietu oprogramowania, etap ten może zostać pominięty.
Dostarczenie (ang. *delivery*) (S_4)	Skopiowanie cyberbroni do docelowego środowiska, np. wykorzystanie najbardziej rozpowszechnionych sposobów dostawy (np. w ramach ataków APT), którymi przykładowo są: zainfekowane załączniki do e-maili, spreparowane lub złośliwie zmodyfikowane oprogramowanie strony internetowej (np. aplety, linki), wstrzyknięcie kodu SQL, zainfekowane nośniki danych podłączane do portów USB.
Uruchomienie i kontrola kodu złośliwego (ang. *cyber execution*) (S_5)	1. Uruchomienie kodu złośliwego (po dostarczeniu cyberbroni do środowiska docelowego), np. w wyniku wykorzystania podatności/luki programowej w aplikacji lub systemie operacyjnym lub zmanipulowania użytkownika systemu docelowego. 2. Instalacja dodatkowego kodu złośliwego, np. koni trojańskich (ang. Remote Access Trojan, RAT), umieszczenie tylnych furtek (ang. backdoor) w systemie docelowym w celu zestawienia stałego kanału komunikacji zainfekowanego środowiska wewnętrznego ofiary z centrum (zewnętrznym środowiskiem) dowodzenia i sterowania oprogramowaniem złośliwym. 3. Kontrola i sterowanie zainfekowanego środowiska, np. eskalacja lub uzyskanie dodatkowych uprawnień, systemowych, doinstalowanie pozostałego lub dodatkowego kodu złośliwego (np. ang. backdoor/trojan/rootkit), modyfikacja system plików, przeglądanie lub modyfikacja systemowych baz danych.
Realizacja celów (*achieve objectives, action*) (S_6)	Podjęcie działań nakierowanych na osiągnięcie pierwotnych celów, np. skopiowanie danych, naruszanie integralności i/lub dostępności danych, uzyskanie dostępu do poczty elektronicznej ofiary w celu wykorzystania jej do głębszej penetracji zakatowanej infrastruktury lub wykorzystanie poczty elektronicznej do dalszego rozprzestrzenienia prowadzonego ataku. W tej fazie nie wyklucza się fizycznej destrukcji infrastruktury organizacji.
Zakończenie ataku i zatarcie śladów (S_7)	Zakończenie ataku, może być połączone z usunięciem lub zamaskowaniem śladów ataku i aktywności kodu złośliwego. Etap opcjonalny, zależny od celów i stopnia zaawansowania technologicznego agresora.

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

3. Podstawowe założenia modelu

Jak już wspomniano na wstępie, podstawą proponowanego w niniejszym artykule stochastycznego modelu jest ogólny cykl życia ataku cybernetycznego (omówiony w poprzednim paragrafie; tab. 1, rys. 1). Na potrzeby prezentowanego modelu przyjmuje się, że nie ma możliwości powrotu z bieżącej fazy do faz poprzednich* oraz pominięcia której kolwiek z faz następnych. Natomiast w modelu uwzględnia się dynamikę procesu ataku, zakładając, że atak może zostać powstrzymany lub przerwany (w ostateczności zakończony) w każdej z faz i w dowolnej chwili, licząc od momentu rozpoczęcia danej fazy ataku. Na rysunku 1 zobrazowano w formie grafu skierowanego możliwe przejścia pomiędzy poszczególnymi fazami cyklu.

W praktyce na jednym cyklu ataku cybernetycznego na daną organizację (przedsiębiorstwo, instytucję, państwo) zazwyczaj się nie kończy. Ataki są powtarzane do skutku, nawet gdyby miało to trwać miesiącami. Jeżeli już atakujący uzyska oczekiwany efekt, to i tak następnie wyznacza sobie nowy cel i rozpoczyna cykl od nowa. Wobec tego zakłada się, że cyberataki mogą być powtarzane, a przerwanie bieżącego ataku skutkuje przystąpieniem przez atakującego do rozpoczęcia nowego cyklu.



RYS. 1. Graf przejść pomiędzy fazami ogólnego cyklu życia ataku cybernetycznego
FIG. 1. Graph of transitions between the phases of the general cyber-attack lifecycle

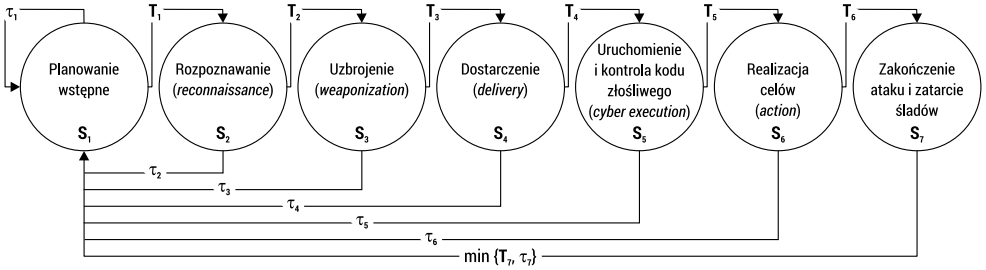
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Przyjmijmy, że $T_n \in [0, +\infty))$ oznacza czas niezbędny do zakończenia z sukcesem fazy S_n ($n = 1, 2, \ldots, 7$). Dalej czasy te nazywać będziemy *niezbędnymi czasami trwania faz*. Zakładamy, że T_n jest zmienną losową o dystrybuancie $F_n(t) = \Pr\{T_n < t\}$ i skończonej wartości oczekiwanej ET_n ($ET_n = \int_0^{+\infty} t dF_n(t) < +\infty$). Niech $\tau_n \in [0, +\infty)$ będzie zmienną losową o dystrybuancie $G_n(t) = \Pr\{\tau_n < t\}$ i skończonej wartości oczekiwanej $E\tau_n$ ($E\tau_n = \int_0^{+\infty} t dG_n(t) < +\infty$) oznaczającą czas, po którym może nastąpić zatrzymanie

* Z wyjątkiem powrotu do fazy , który symbolizuje i oznacza rozpoczęcie nowego cyklu ataku.

(przerwanie) ataku, licząc od momentu rozpoczęcia fazy S_n. W tym miejscu, w celu uogólnienia, uczynimy założenie, że bieżący cykl ataku może zakończyć się również w fazie planowania wstępnego. W związku z powyższym czas przebywania w każdej fazie S_n jest równy $\beta_n = \min\{T_n, \tau_n\}$ ($n = 1, 2, \ldots, 7$).

Zgodny z przyjętymi założeniami graf przejść pomiędzy stanami przedstawiono na rysunku 2. Powrót do stanu S_1 symbolizuje rozpoczęcie nowego cyberataku – nowego cyklu ataku cybernetycznego. Łuki grafu przejść pomiędzy fazami cyklu cyberataku opisano czasami, po których może nastapić przejście do fazy następnej lub w ostateczności zakończenie lub przerwanie ataku.



RYS. 2. **Graf ogólnego cyklu życia ataku cybernetycznego z czasami przebywania w fazach**
FIG. 2. **Graph of the general cyber-attack lifecycle with staying time in phases**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

W tym miejscu dodatkowo przyjmuje się założenie, że niezbędne czasy trwania poszczególnych faz są niezależne stochastycznie, tzn. zmienne losowe $T_1, \ldots, T_7$ są niezależnymi zmiennymi losowymi. Ponadto niech zmienne losowe $\tau_1, \ldots, \tau_7$ będą również niezależnymi zmiennymi losowymi. Zakłada się również niezależność stochastyczną zmiennych losowych T_n i τ_n ($n = 1, 2, \ldots, 7$).

4. Czas trwania pojedynczego cyklu ataku cybernetycznego

Przyjmijmy, że $\alpha_n \in \{0,1\}$ dla każdego $n = 1, 2, \ldots 7$ jest binarną zmienną losową[**] – taką, że $\alpha_n = 1$, gdy zachodzi zdarzenie $\{T_n < \tau_n\}$ oraz $\alpha_n = 0$, gdy zachodzi $\{T_n \geq \tau_n\}$. Niech $\Theta \in [0, +\infty)$ oznacza czas trwania pojedynczego cyklu ataku cybernetycznego.

Wobec przyjętych oznaczeń i założeń czas trwania pojedynczego cyklu ataku cybernetycznego Θ można zapisać następująco:

$$\Theta = \beta_1 + \alpha_1 \cdot \beta_2 + \alpha_1 \cdot \alpha_2 \cdot \beta_3 + \alpha_1 \cdot \alpha_2 \cdot \alpha_3 \cdot \beta_4 + \ldots + \alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_6 \cdot \beta_7 \qquad (1)$$

gdzie $\beta_n = \min\{T_n, \tau_n\}$.

[**] Indykatorem zdarzenia.

Ze wzoru (1) wynika, że czas trwania pojedynczego cyklu życia ataku cybernetycznego Θ jest sumą zależnych zmiennych losowych – czasów przebywania w poszczególnych fazach cyklu ataku zależnych od zachowania się procesu ataku w fazach poprzedzających daną fazę.

Niech $\overline{F_n}(t) = 1 - F_n(t)$ oraz $\overline{G_n}(t) = 1 - G_n(t)$. Zatem, na podstawie przyjętych założeń o niezależności zmiennych losowych T_n i τ_n, dystrybuanta zmiennej losowej $\beta_n = \min\{T_n, \tau_n\}$ wyraża się wzorem:

$$B_n(t) = \Pr\{\beta_n < t\} = Pr\{\min\{T_n, \tau_n\} < t\} = 1 - \overline{F_n}(t) \cdot \overline{G_n}(t) \tag{2}$$

Wobec tego wartość oczekiwana zmiennej losowej β_n przyjmuje postać:

$$E\beta_n = \int_0^{+\infty} \overline{F_n}(t) \cdot \overline{G_n}(t)\, dt \tag{3}$$

Wartość oczekiwana zmiennej losowej α_n wynosi:

$$E\alpha_n = 1 \cdot Pr\{T_n < \tau_n\} + 0 \cdot Pr\{T_n \geq \tau_n\}$$

Stąd ostatecznie:

$$E\alpha_n = \int_0^{+\infty} F_n(t)\, dG_n(t) = \int_0^{+\infty} \overline{G_n}(t)\, dF_n(t) \tag{4}$$

Z własności wartości oczekiwanej zmiennej losowej [8] wynika, że wartość oczekiwana czasu trwania cyklu ataku cybernetycznego Θ przyjmuje postać:

$$E\Theta = E\beta_1 + E\alpha_1\beta_2 + E\alpha_1\alpha_2\beta_3 + ... + E\alpha_1\alpha_2 \cdot ... \cdot \alpha_6\beta_7$$

Na podstawie przyjętych założeń o niezależności zmiennych losowych T_n i τ_n wartość oczekiwanej długości cyklu ataku cybernetycznego Θ ostatecznie możemy zapisać jako:

$$E\Theta = E\beta_1 + E\alpha_1 \cdot E\beta_2 + E\alpha_1 \cdot E\alpha_2 \cdot E\beta_3 + ... + E\alpha_1 \cdot E\alpha_2 \cdot ... \cdot E\alpha_6 \cdot E\beta_7 \tag{5}$$

gdzie $E\beta_n (n = 1, 2, ..., 7)$ dla dane jest wzorem (3), a dla $E\alpha_n$ – wzorem (4).

5. Proces ataku jako stochastyczny proces regenerujący się

Załóżmy, że $\{X(t), t \in [0, +\infty\}$ jest procesem stochastycznym przyjmującym wartości ze zbioru $\mathbf{S} = \{1, 2, ..., 7\}$, którego kolejne elementy są numerami faz pojedynczego cyklu ataku – odpowiednio: $S_1, S_2, ..., S_7$. Stąd też, długość Θ czasu trwania pojedynczego cyklu ataku cybernetycznego jest chwilą zatrzymania procesu $\{X(t), t \leq \Theta\}$.

W celu dalszych rozważań oznaczmy przez Θ_k, $k = 1, 2, ...$ następujące po sobie cykle ataku cybernetycznego zdefiniowane wzorem (1), a przez $X_k(t)$ – odpowiadający każdemu cyklowi o numerze k proces $X(t)$.

Z wcześniej poczynionych założeń wynika, że ciąg zmiennych losowych $\{\Theta_k\}_{k\geq 1}$ możemy uważać za ciąg niezależnych zmiennych losowych. Przyjmując założenie, że Θ_k mają jednakowy rozkład prawdopodobieństwa z wartością oczekiwana równą $E\Theta$, daną wzorem (5) możemy przyjąć, że pary $(X_k(t), \Theta_k)$, k ∈ {1, 2, ...} są niezależne i o jednakowym rozkładzie. Niech $Y(t)$, $t \in [0, +\infty)$ będzie procesem stochastycznym przyjmującym wartości ze zbioru ***S*** i zdefiniowanym następująco:

$$Y(t) = X_k(t - t_k),\ t_{k-1} \leq t < t_k,\ t_k = \Theta_1 + \Theta_2 + ... + \Theta_k,\ t_0 = 0,\ k \geq 1 \tag{6}$$

Tak zdefiniowany proces $Y(t)$ jest procesem regenerującym się [9], a chwile t_k są momentami regeneracji, w których rozpoczyna się nowy cykl ataku cybernetycznego. W tym przypadku przedziały $[t_{k-1}, t_k)$ są okresami regeneracji procesu (6).

Biorąc pod uwagę założenie powtarzalności (cykli) ataków cybernetycznych, będziemy dalej rozpatrywać ciąg cykli $\{(X_k(t), \Theta_k)\}_{k\geq 1}$ jako model cyklicznego ataku cybernetycznego – ciągłego ataku powtarzanego wg. przyjętego cyklu życia ataku cybernetycznego. Rozpatrywane tutaj cykle $(X_k(t), \Theta_k)$ na podstawie wcześniej przyjętych założeń są stochastycznie niezależne oraz mają rozkłady prawdopodobieństwa stochastycznie równoważne zmiennej losowej Θ. Należy tutaj zauważyć, że z założeń o możliwości przerwania ataku cybernetycznego w dowolnej fazie i, w konsekwencji tego, z konstrukcji czasu Θ trwania pojedynczego cyklu ataku cybernetycznego (1) wynika, że proces $Y(t)$ nie zawsze będzie przechodził przez wszystkie numery faz[***].

6. Stacjonarny rozkład prawdopodobieństwa procesu ataku

Naszym podstawowym zadaniem będzie znalezienie rozkładu stacjonarnego stochastycznego procesu regenerującego się $\{Y(t) \in \{1, 2, ..., 7\}, t \geq 0\}$ określonego przez (6), sprowadzające się do wyznaczenia granic:

$$\lim_{t\to\infty} Pr\{Y(t) = n\} = P_n \tag{7}$$

gdzie n jest numerem fazy S_n ($n = 1, 2, ..., 7$) ataku cybernetycznego.

Z węzłowego twierdzenia odnowy oraz twierdzenia Smitha [7], [9] wynika, że granicę (7) można wyznaczyć z zależności:

$$\lim_{t\to\infty} Pr\{Y(t) = n\} = \frac{1}{E\Theta}\int_0^{+\infty} Pr\left\{\left\{Y(t) = n\right\} \cap \left\{\Theta \geq t\right\}\right\} dt \tag{8}$$

W tym celu w pierwszej kolejności musimy wyznaczyć prawdopodobieństwo tego, że w chwili $t \geq 0$ proces ataku cybernetycznego jest w fazie S_n oraz że atak nie zakończył się do chwili t, tzn. $\Pr\left\{\left\{Y(t) = n\right\} \cap \left\{\Theta \geq t\right\}\right\}$. Zauważmy, że prawdopodobieństwo to jest następujące:

[***] Wszystkie fazy cyklu ataku.

$$
\begin{aligned}
Pr\left\{\left\{Y(t)=n\right\}\cap\left\{\Theta\geq t\right\}\right\} &= Pr\left\{\left\{\sum_{i=1}^{n-1}d_i<t\right\}\cap\left\{\sum_{i=1}^{n}d_i\geq t\right\}\right\}= \\
&= 1-Pr\left\{\left\{\sum_{i=1}^{n-1}d_i\geq t\right\}\cup\left\{\sum_{i=1}^{n}d_i<t\right\}\right\}= \\
&= 1-Pr\left\{\sum_{i=1}^{n}d_i<t\right\}-Pr\left\{\sum_{i=1}^{n-1}d_i\geq t\right\}= \\
&= Pr\left\{\sum_{i=1}^{n}d_i\geq t\right\}-Pr\left\{\sum_{i=1}^{n-1}d_i\geq t\right\}
\end{aligned}
\tag{9}
$$

gdzie $\Theta = d_1 + ... + d_7$, $d_1 = \beta_1$, $d_i = \alpha_1 \cdot ... \cdot \alpha_{i-1} \cdot \beta_i$ dla $i = 2, ..., 7$.

Zauważmy, że na podstawie wzoru (8) otrzymujemy:

$$
\int_0^{+\infty} Pr\left\{\left\{Y(t)=n\right\}\cap\left\{\Theta\geq t\right\}\right\} = \int_0^{+\infty} Pr\left\{\sum_{i=1}^{n}d_i\geq t\right\}-Pr\left\{\sum_{i=1}^{n-1}d_i\geq t\right\}dt
$$

gdzie $\Theta = d_1 + ... + d_7$, $d_1 = \beta_1$, $d_i = \alpha_1 \cdot ... \cdot \alpha_{i-1} \cdot \beta_i$ dla $i = 2, ..., 7$.

W powyższym wyrażeniu całka $\int_0^{+\infty} Pr\left\{\sum_{i=1}^{n}d_i\geq t\right\}dt$ określa nam wartość oczekiwaną [7] sumy zmiennych losowych $\sum_{i=1}^{n}d_i$. Z własności wartości oczekiwanych [8] wynika natomiast, że $E\left\{\sum_{i=1}^{n}d_i\right\} = \sum_{i=1}^{n}Ed_i$. Stąd też możemy zapisać:

$$
\int_0^{+\infty} Pr\left\{\left\{Y(t)=n\right\}\cap\left\{\Theta\geq t\right\}\right\} = E\left\{\sum_{i=1}^{n}d_i\geq t\right\}-E\left\{\sum_{i=1}^{n-1}d_i\geq t\right\} = Ed_n
$$

gdzie $Ed_1 = E\beta_1$, $Ed_n = E\alpha_1 \cdot ... \cdot \alpha_{n-1} \cdot \beta_n$ dla $n \geq 2$.

Wobec powyższego oraz na podstawie (5) granica ostatecznie (9) jest następująca dla każdego $n = 1, 2, ..., 7$:

$$
\lim_{t\to\infty} Pr\{Y(t)=n\} = P_n = \frac{Ed_n}{E\Theta} = \frac{E\alpha_1 \cdot \ldots \cdot E\alpha_{n-1} \cdot E\beta_n}{E\Theta}
\tag{10}
$$

gdzie $E\alpha_1$, $E\alpha_{n-1}$, $E\beta_n$, $E\Theta$ są określone odpowiednio wzorami (3), (4) i (5).

7. Podsumowanie

Przedstawiony w pracy stochastyczny model ogólnego cyklu życia ataku cybernetycznego bazuje na opisie ataku z [1], wyróżniającego się na tle innych publikowanych w literaturze [2, 3, 4, 5, 6] definicji cyklu życia ataku cybernetycznego dodanymi dwoma fazami: identyfikacji i definicji celów, które traktowane są jako planowanie

wstępne ataku oraz zakończenie ataku połączone z zatarciem śladów aktywności agresora. Należy tutaj wskazać także, że w odróżnieniu od dotychczasowego ujęcia problemu, prezentowanego przez innych badaczy [2, 3, 4, 5, 6], w przyjętym w pracy [1] cyklu życia ataku czynności takie, jak: uruchomienie, ewentualna instalacja kodu złośliwego oraz dowodzenie, kierowanie i sterowanie występują jako jedna faza.

Z uwagi na to, że do tej pory w dostępnych źródłach [1, 10, 11] nie publikowano stochastycznego modelu cyklu życia ataku przy założeniu dowolnego ciągłego rozkładu prawdopodobieństwa przebywania cyklu ataku w poszczególnych fazach, w pracy wypełnia się tę lukę, bazując na ogólnym cyklu życia ataku. Podobnie jak w pracach [10, 11] zakłada się możliwość przerwania ataku w trakcie przebywania agresora w dowolnej fazie ataku. W praktyce przerwanie ataku może nastąpić nie tylko z woli cyberprzestępcy, ale również, a może przede wszystkim, z powodu działającego systemu cyberobrony.

Na bazie przedstawionego modelu wyliczone charakterystyki probabilistyczne, takie jak stacjonarne prawdopodobieństwa przebywania procesu ataku w poszczególnych fazach czy wartości oczekiwane czasów trwania poszczególnych faz można wykorzystać na potrzeby procesu szacowania ryzyka i zarządzania bezpieczeństwem organizacji i świadczonych e-usług.

Należy tutaj wskazać, że z praktycznego punktu widzenia do oszacowania ww. charakterystyk potrzeba i wystarcza znajomość wartości oczekiwanych poszczególnych czasów oraz oszacowanie prawdopodobieństwa sukcesu zakończenia poszczególnych faz cyklu ataku.

Literatura

1. Hoffmann R. Ogólny cykl życia ataku cybernetycznego i jego markowowski model. *Ekonomiczne Problemy Usług*. 2018; vol. 1, 2/2018(131): 121-130.
2. Coleman KGJ. *Aggression in Cyberspace*. In: Jasper S, ed. *Conflict and Cooperation in the Global Commons: A Comprehensive Approach for International Security*. Washington DC: Georgetown University Press; 2012; 105-119.
3. Hahn A, Thomas RK, Lozano I, Cardenas A. A multi-layered and kill-chain based security analysis framework for cyber-physical systems. *International Journal of Critical Infrastructure Protection*. 2015; 11: 39-50.
4. Hutchins EM, Cloppert MJ, Amin RM. *Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains*. In Ryan J, ed. *Leading Issues in Information Warfare and Security Research*, vol. 1. Reading, UK: Academic Publishing International Ltd; 2011; 78-104.
5. Khan MS, Siddiqui S, Ferens K. *A Cognitive and Concurrent Cyber Kill Chain Model*. In: Daimi K, ed. *Computer and Network Security Essentials*. Cham, Switzerland: Springer; 2018; 585-602.
6. Spring JM, Hatleback E. Thinking about intrusion kill chains as mechanisms. *Journal of Cybersecurity*. 2017; 3(3): 185-197.

7. Klimow GP. *Procesy obsługi masowej*. Warszawa: WNT; 1979.
8. Beichelt F. *Stochastic Processes in Science, Engineering and Finance*. New York: Taylor & Francis Group, LLC; 2006.
9. Kowalenko IN, Kuzniecow NJ, Szurienkow WM. *Procesy stochastyczne. Poradnik*. Warszawa: PWN; 1989.
10. Hoffmann R. Markowowskie modele cykli życia ataku cybernetycznego. *Roczniki Kolegium Analiz Ekonomicznych SGH*. 2019; 54: 303-317.
11. Hoffmann R. *Markov Models of Cyber Kill Chains with Iterations*. Materiały z: International Conference on Military Communications and Information Systems – ICMCIS 2019, Montenegro, Budva 2019.

Streszczenie

W pracy oparto się na opublikowanym w literaturze ogólnym cyklu życia ataku cybernetycznego składającym się z następujących faz: identyfikacja i definicja celów ataku – planowanie wstępne, rozpoznanie, uzbrojenie, dostarczenie, uruchomienie i kontrola kodu złośliwego, realizacja celów, zakończenie ataku i zatarcie śladów. W odróżnieniu od modelu źródłowego przyjęto założenie dowolnych ciągłych rozkładów prawdopodobieństwa czasów trwania poszczególnych faz cyklu życia ataku. Ponadto w modelu uwzględniono powtarzalność ataków cybernetycznych. Na tej podstawie wyznaczono stacjonarne prawdopodobieństwa przebywania procesu ataku w poszczególnych fazach.

Słowa kluczowe: atak cybernetyczny, cykl życia ataku cybernetycznego, stochastyczny proces regenerujący się, stochastyczny model, proces ataku, prawdopodobieństwo stacjonarne

Summary
Stochastic model of the general cyber-attack life cycle

The proposed model bases on the description of the general cyber-attack life cycle already published in the literature which consist of the following phases: identification and definition of attack targets and goals – initial planning, reconnaissance, weaponization, delivery, cyber execution and command & control, achieve objectives, ending a cyber-attack and removing traces attacker's activities.

The presented model is distinguishable from the source model by the assumption of any continuous probability distribution of the durations of the cyber-attack life cycle phases was assumed. Additionally, the presented model includes the assumption of the repeatability of cyber-attacks. On this basis, the stationary probabilities of staying of the attack process in individual phases were determined.

Keywords: cyber-attack, stochastic model, cyber-attack life cycle, regenerative stochastic process, attack process, stationary probability

DOI: 10.24427/978-83-66391-28-4_3

Układ żył w palcu i odcisk palca w multimodalnym systemie biometrycznym

Maciej SZYMKOWSKI[*]
Khalid SAEED[*]

1. Wstęp

Na przestrzeni ostatnich dziesięciu lat możemy zaobserwować znaczący wzrost zainteresowania biometrycznymi metodami rozpoznawania tożsamości człowieka. Biometria przestała być zagadnieniem kojarzącym się z filmami science-ficton, a stała się elementem naszego codziennego życia. Skanery odcisku palca, tęczówki czy też twarzy są powszechnie wykorzystywane, także w zminiaturyzowanej formie – np. w smartfonach.

Z wykorzystywaniem cech biometrycznych wiąże się jednak także pewne niebezpieczeństwo. Otóż część z nich można dosyć łatwo spreparować – np. odcisk palca może zostać wykonany z modeliny. Bardzo często, pomimo wprowadzanych rozwiązań zaradczych, systemy biometryczne dają się oszukać i uznają, że przedstawiona sfałszowana próbka jest zgodna ze wzorcem przechowywanym w bazie danych. Jednym ze sposobów przeciwdziałania tego rodzaju oszustwom są multimodalne systemy biometryczne. Są to bowiem rozwiązania, które do rozpoznawania człowieka wykorzystują więcej aniżeli pojedynczą cechę biometryczną. Atakujący musiałby w związku z tym spreparować co najmniej dwie cechy, co z kolei powoduje, że włamanie do systemu zajęłoby znacząco więcej czasu. Istnieje również bardzo duże prawdopodobieństwo, że system odrzuciłby przynajmniej jedną ze sfałszowanych próbek, co z kolei doprowadziłoby do nierozpoznania tożsamości użytkownika.

W literaturze możemy znaleźć wiele podobnych, zróżnicowanych metod, także odnoszących się do rozwiązań multimodalnych. Jednakże większość z nich łączy w sobie najczęściej pojedynczą cechę fizjologiczną oraz pojedynczą cechę behawioralną. Doskonałym przykładem może być fuzja odcisku palca i szybkości pisania na klawiaturze. Jednakże z punktu widzenia autorów taki system jest bardzo uciążliwy w użytkowaniu. Zwróćmy bowiem uwagę na to, że nie jesteśmy w stanie dokonać akwizycji różnych danych w jednym i tym samym momencie. Potrzebujemy oddzielnie przekazać odcisk palca, a następnie wprowadzić odpowiednie dane przy użyciu klawiatury.

W ramach niniejszej pracy zaprezentowany został multimodalny system biometryczny, który rozpoznaje użytkownika przy pomocy dwóch cech fizjologicznych, tj. układu żył w palcu oraz odcisku palca. Obydwie cechy zostały wybrane ze względu

[*] Politechnika Białostocka

na łatwość ich pobierania (do pobrania układu żył palca autorzy przygotowali własne urządzenie), jak również wysoki poziom skuteczności rozpoznawania tożsamości człowieka. Ponadto, autorzy opracowali schemat systemu, który byłby w stanie jednocześnie pobrać obydwie cechy biometryczne. Znacząco ułatwia to, z punktu widzenia użytkownika, proces pobierania próbek i powoduje, że staje się on mniej uciążliwy.

2. Aktualny stan wiedzy

Współcześnie istnieje wiele rozwiązań biometrycznych skorelowanych z odciskiem palca czy układem żył. Jednakże w przypadku drugiej z rzeczonych cech najczęściej będziemy mieli z nią do czynienia w kontekście dłoni. Bardzo rzadko badacze zajmują się układem żył w palcu pomimo tego, że gwarantuje on również wysoki poziom skuteczności rozpoznania człowieka oraz jest bardzo prosty w akwizycji. Warto również w tym miejscu podkreślić, co spowodowało wzrost zainteresowania obiema cechami. Otóż obie są trudne do spreparowania (tym bardziej, że coraz częściej stosowane są czujniki badające żywotność poszczególnych cech, a nie tylko dokonujące akwizycji danych), jak również gwarantują bardzo wysoki poziom rozróżnialności. W przypadku odcisków przeprowadzono badania, które jednoznacznie wykluczyły możliwość wystąpienia dwóch idealnie takich samych odcisków palców (nawet w przypadku bliźniaków jednojajowych) [1].

Naszą analizę rozpoczniemy od odcisków palców. Pierwszym interesującym algorytmem jest rozwiązanie, które zostało zaproponowane w ramach artykułu [2]. W tej pracy autorzy zwrócili uwagę na bardzo istotny i ciekawy problem. Mianowicie wskazują oni na to, że zniekształcenia elastyczne odcisków palców mogą mieć bardzo duży wpływ na ich prawidłowe rozpoznawanie oraz błędy systemów biometrycznych. Rozwiązanie, które zostało przez nich zaproponowane, polega na zastosowaniu głębokich sieci neuronowych w celu detekcji i późniejszego usunięcia zniekształceń. Wykrycie rzeczonych elementów odbywa się w oparciu o estymację pewnych parametrów wskazujących na zniekształcenia. Jest to nowatorska metoda, która może być wykorzystywana w ramach realnych zastosowań. W pracy wskazano także bazy, które zostały wykorzystane w trakcie badań – Tsinghua DB oraz FVC2004 DB1.

Drugą interesującą pracą dotyczącą odcisków palców jest [3]. W ramach tej publikacji ponownie wykorzystano techniki uczenia głębokiego do klasyfikacji odcisków palców. W tym przypadku nie użyto jednak klasycznego podejścia bazującego na punktach charakterystycznych (minucjach), lecz oparto się o pewne struktury i kierunki rozchodzenia się poszczególnych krawędzi odcisków palców. Autorzy skorzystali w swoim podejściu z sieci neuronowej zbudowanej na mocy modelu regresji softmax oraz uczenia nadzorowanego. Wyniki zostały wygenerowane na podstawie bazy NIST-DB4.

W badaniach oraz rozwiązaniach przemysłowych najczęściej stosuje się podejście oparte o minucje. Pewnym problemem, z którym zmagają się zróżnicowane zespoły naukowe, jest efektywny sposób opisu takich punktów. Interesujące algorytmy zostały zaprezentowane w pracach [4, 5]. W pierwszej z nich [4] podkreślono bardzo ważną rolę algorytmów grafiki komputerowej oraz metody Crossing Number (CN) do detekcji punktów charakterystycznych odcisków palców. W pracy wskazano jednak, że z użyciem tychże metod można także wykryć bardzo dużo fałszywych minucji, co może znacząco wpłynąć na osiąganą skuteczność systemu biometrycznego. Druga z prac [5] skupia się na wskazaniu dwóch głównych problemów, jakie są obserwowane w ramach metod opartych o punkty charakterystyczne odcisku palca. Mianowicie są to: wybór odpowiedniej miary podobieństwa pomiędzy próbkami oraz odpowiednia interpretacja poziomu podobieństwa dwóch odcisków palców. Autorzy wskazali, że najefektywniej system działał wtedy, gdy wektor cech składał się z 17 elementów oraz przy użyciu metody wektorów wspierających (SVM).

Drugą cechą, którą analizujemy w ramach proponowanej pracy jest układ żył w palcu. W przypadku tegoż elementu istnieją jednak dwa główne aspekty, nad którymi musimy się pochylić. Pierwszym z nich jest sposób konstrukcji urządzenia, które pozwoli nam na akwizycję danych, natomiast drugim jest metodyka konstrukcji algorytmu przetwarzania obrazów i ekstrakcji cech. W literaturze praktycznie nie spotyka się metod odnoszących się do układu żył w palcu. Znacznie częściej możemy zaobserwować algorytmy bazujące na tejże cesze, aczkolwiek występującej w dłoni. Pierwszą pracą dotyczącą takiego systemu jest [6]. W niej rzeczywiście skorzystano bezpośrednio z układu żył w palcu. W ramach metody nie opisano jednak, w jaki sposób następowała akwizycja danych, skupiono się głównie na drugim aspekcie, czyli algorytmie przetwarzania obrazu. W tym przypadku bazował on na podejściu *Local Binary Patterns* (LBP).

Następną interesującą pracą jest system multimodalny, który łączy układ żył w dłoni oraz jej geometrię [7]. W ramach wstępnego przetwarzania obrazu użyto podstawowych metod przetwarzających obraz – np. filtru Gaussa czy filtrów górnoprzepustowych. Ekstrakcja cech z kolei rozpoczynała się od zastosowania filtru uwypuklającego poszukiwane atrybuty. W przypadku tegoż podejścia skorzystano z połączenia wierszowego oraz kolumnowego filtra VPE (*Vascular Pattern Extraction*), w skład, którego wchodzą elementy usuwające szum oraz uwydatniające poszczególne cechy.

Interesujące rozwiązanie zostało przedstawione także w artykule [8]. W tym przypadku autorzy opracowali samodzielnie urządzenie, które składało się z: diod bliskiej i dalekiej podczerwieni, specjalizowanej kamery pozwalającej na pobieranie obrazów podczerwonych, a także szkła dyfuzyjnego. Sam algorytm ponownie został skonstruowany w oparciu o podstawowe metody przetwarzania obrazów: konwersję do skali szarości, modyfikację kontrastu, binaryzację oraz metody morfologiczne, a także algorytm Sobela oraz szkieletyzację.

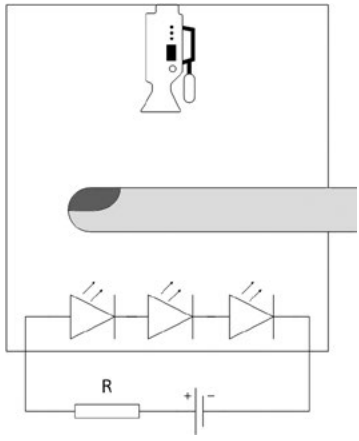
3. Proponowane rozwiązanie

W ramach proponowanego rozwiązania autorzy postanowili skorzystać z dwóch algorytmów opublikowanych wcześniej [9, 10]. Następnym krokiem było udoskonalenie obydwu podejść poprzez wprowadzenie modyfikacji do metod wstępnego przetwarzania obrazów oraz ekstrakcji cech. Uwzględnione zmiany pozwoliły na osiągnięcie znacząco lepszych rezultatów pod kątem wyodrębniania elementów wchodzących w skład wektora cech. Nasz opis rozpoczniemy od informacji skorelowanych z układem żył w palcu, a następnie omówimy algorytm przetwarzania odcisków palców. Finalnym elementem niniejszego podrozdziału będą informacje dotyczące sposobu konstrukcji systemu multimodalnego.

3.1. Algorytm przetwarzania układu żył w palcu

Algorytm przetwarzania układu żył w palcu wykorzystany w ramach niniejszej pracy bazował w dużej mierze na propozycji, która została opisana w publikacji [9]. Jednakże zanim zostanie on zaprezentowany, warto odnieść się do sposobu konstrukcji urządzenia wykorzystywanego do pobierania tejże cechy.

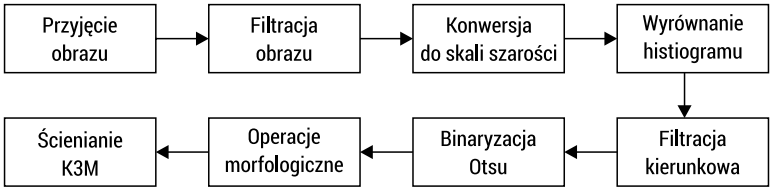
Wspomniane urządzenie składa się z trzech diod LED światła podczerwonego (o długości fali 840-870 nm), kamery Tracer Prospecto Cam umieszczonej bezpośrednio nad palcem oraz obudowy, która ma za zadanie zaciemnić otoczenie. Schemat koncepcyjny danego modułu został zaprezentowany na rysunku 1.



RYS. 1. Schemat urządzenia do akwizycji układu żył w palcu
FIG. 1. Device scheme for finger veins acquisition

ŹRÓDŁO: [9].
SOURCE: [9].

Drugą częścią naszego rozwiązania jest algorytm przetwarzania obrazu układu żył w palcu oraz ekstrakcji jego cech. To rozwiązanie zostało przygotowane w oparciu o język programowania Java oraz matematyczny framework zaimplementowany przez autorów. Na rysunku 2 przedstawiony został pełny schemat blokowy tego rozwiązania.



RYS. 2. Schemat blokowy proponowanego rozwiązania
FIG. 2. Block scheme of the proposed solution

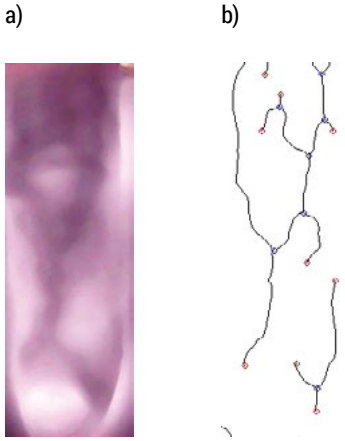
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Proponowany algorytm rozpoczyna się od przycięcia obrazu – jest to spowodowane niedoskonałością wykonanego urządzenia, które poza samym układem żył w palcu pobiera także dużo informacji z jego otoczenia. Po wykonaniu tejże operacji następuje filtracja obrazu, która jest realizowana z użyciem dwóch filtrów – medianowego oraz rozmywającego. Użycie obydwu tych filtrów jednocześnie pozwoliło nam na osiągnięcie znacząco lepszego oczyszczenia obrazu z drobnych zniekształceń, które pozostały po wykonanej akwizycji.

Kolejnym krokiem była konwersja do skali szarości. W tym przypadku bazowaliśmy na kanale zielonym, który stał się swoistym standardem w ramach rozwiązań dotyczących układu żył (czy to w dłoni, palcu czy siatkówce oka) [11-13]. Wykonanie tej operacji pozwoliło nam na uwypuklenie układu żył. Kolejnym krokiem było wyrównanie histogramu. Operacja ta umożliwiła nam zaobserwowanie detali, które były niewidoczne po nałożeniu poprzedniego algorytmu.

Kolejne kroki były związane z dalszą poprawą jakości naszego obrazu. Zastosowaliśmy bowiem procedurę binaryzacji, która dokonała finalnej konwersji obrazu do postaci czarno-białej. Pozwoliło to nam na oddzielenie układu żył od tła. Następne algorytmy, czyli inwersja kolorów, morfologiczne zamknięcie, a także filtracja medianowa, zostały użyte w celu usunięcia dodatkowych zniekształceń, jakie były widoczne po procesie binaryzacji.

Ostatnim krokiem było wykonanie szkieletyzacji i ekstrakcja cech. W przypadku pierwszej z rzeczonych operacji wykorzystaliśmy doskonale nam znany algorytm K3M [14], natomiast w drugiej z procedur użyliśmy algorytmu Crossing Number (CN) [15]. Finalny wynik został zaprezentowany na rysunku 3b, natomiast oryginalny obraz, jaki otrzymaliśmy na początku, przedstawiony został w ramach rysunku 3a.



RYS. 3. **Układ żył uzyskany po akwizycji danych (a) oraz obraz po przetworzeniu i ekstrakcji cech (b)**
FIG. 3. **Finger veins after image acquisition (a) and image after processing and feature extraction (b)**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

3.2. Algorytm przetwarzania odcisku palca

Drugim, niezwykle ważnym algorytmem, który zastosowano w ramach niniejszej pracy, było rozwiązanie dotyczące przetwarzania i ekstrakcji cech odcisku palca. W tym przypadku skorzystaliśmy bezpośrednio z rozwiązania opisanego w ramach publikacji [10]. Wprowadziliśmy jedynie drobną korektę, która pozwoliła nam na osiągnięcie lepszego rezultatu w kontekście wstępnego przygotowania obrazu. Poszczególne kroki algorytmu skorelowanego z odciskami palców przedstawiono w ramach algorytmu 1.

Algorytm 1. Pseudokod algorytmu przetwarzania i ekstrakcji cech odcisku palca

```
algorithm fingerprint {
    image = wczytaj_obraz(ściezka)
    image.przytnij(ROI)
    image.binaryzacja_Otsu()
    image.filtruj_obraz(Filtr.MEDIANOWY)
    image.filtruj_obraz(Filtr.ROZMYWAJACY)
    image.szkieletyzacja(Algorytm.K3M)
    image.ekstrakcja_cech(Metoda.SIECI_NEURONOWE)
    image.potwierdz_cechy(Metoda.CN)
}
```

Pierwszym krokiem naszego rozwiązania jest wczytanie obrazu spod zadanej ścieżki. W kolejnym etapie następuje przycięcie obrazu do zadanego wcześniej regionu zainteresowania (ROI). Ten element jest realizowany ze względu na niedoskonałość

metody akwizycji obrazu. Po pobraniu odcisku palca jesteśmy bowiem w stanie zaobserwować dodatkowe białe elementy, które nie należą do odcisku palca i mogą być usunięte przed dalszymi etapami.

Następnym algorytmem zastosowanym w ramach naszej, solucji jest metoda binaryzacji Otsu. W trakcie eksperymentów autorzy dokonali porównania wielu zróżnicowanych procedur, m.in. Bernsena, Otsu czy Niblacka, jednakże wyniki badań pokazały, że odcisk palca jest najlepiej odwzorowany (tzn. nie obserwujemy straty danych oraz poszczególne jego elementy są prezentowane w sposób klarowny) z użyciem metody Otsu. Jej zastosowanie umożliwiło nam dokładne odseparowanie odcisku palca od tła. W następnych dwóch etapach wykonaliśmy procedury filtracji (medianowej oraz rozmywającej), które pozwoliły nam na usunięcie zbędnych elementów widocznych po procedurze binaryzacji.

RYS. 4. Oryginalny obraz pobrany z użyciem skanera odcisków palców (a) oraz obraz po przetworzeniu i ekstrakcji cech (b)

FIG. 4. Original image acquired with fingerprint scanner (a) and image after processing and feature extraction (b)

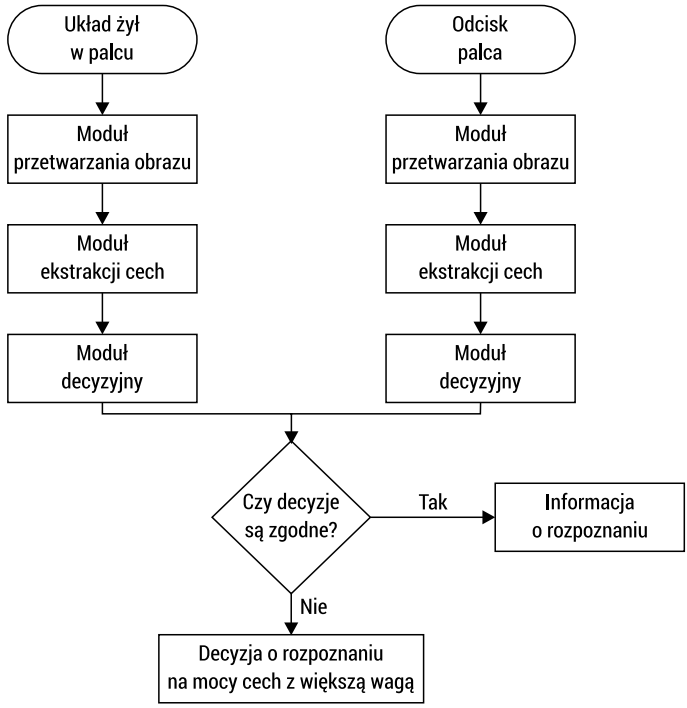
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Po zakończeniu tego etapu, w celu redukcji danych, użyta została metoda szkieletyzacji, czyli reprezentacji odcisku palca za pomocą linii o grubości pojedynczego piksela. Procedura ta jest niezmiernie ważna, gdyż pozwala na finalne przygotowanie odcisku palca do ekstrakcji cech. W kontekście wyodrębniania punktów szczególnych postanowiliśmy skorzystać z dwóch metod. Pierwsza z nich została oparta o zastosowanie sieci neuronowych. Mianowicie zdefiniowaliśmy zróżnicowane możliwe minucje w formie maski o wymiarach 3x3 i na podstawie tego zbioru wykonaliśmy uczenie sieci. Sama sieć składała się z warstwy wejściowej (9 węzłów), dwóch warstw ukrytych (każda po 25 węzłów) oraz warstwy wyjściowej (3 węzły). Nasza sieć zwracała nam bowiem informację, czy analizowany piksel jest w rzeczywistości minucją czy też nie. Aby jednak mieć całkowitą pewność, że analiza została wykonana w sposób poprawny, ostatnim etapem naszego algorytmu było zastosowanie metody CN, która to ma za zadanie przeanalizować ponownie wszystkie odnalezione punkty

szczególne. W ten sposób unikamy sytuacji, w której wśród znalezionych punktów znajdują się fałszywe minucje. Na podstawie danych zwróconych przez algorytm CN wykonujemy konstrukcję wektora cech, który w kolejnym kroku jest przekazywany do dalszego przetwarzania w multimodalnym systemie biometrycznym. Obraz oryginalny oraz wynik działania opisanej procedury zostały przedstawione na rysunku 4.

3.3. System multimodalny łączący układ żył oraz odcisk palca

W ramach niniejszej podsekcji przedstawiony zostanie główny element naszej pracy, którym jest multimodalny system biometryczny. Jak wspomnieliśmy wcześniej, został on przygotowany w celu zapewnienia większego bezpieczeństwa potencjalnym użytkownikom. Konstrukcja składająca się z dwóch cech biometrycznych pozwala bowiem na zwiększenie pewności co do tożsamości użytkownika, jak również umożliwia zmniejszenie prawdopodobieństwa tego, że osoba posiadająca sfałszowane cechy biometryczne uzyska dostęp do chronionych danych.



RYS. 5. Schemat multimodalnego systemu biometrycznego wykorzystywanego w ramach niniejszych badań

FIG. 5. Multimodal biometrics system scheme that was used during experiments

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Zaimplementowany przez nas system biometryczny został przedstawiony w ramach rysunku 5. Składa się on z dwóch podstawowych modułów – pierwszy z nich odpowiada za przetwarzanie obrazu układu żył w palcu i ekstrakcję jego podstawowych cech, natomiast drugi realizuje analogiczne zadania w stosunku do odcisków palca.

Zasada działania systemu multimodalnego jako całości jest nastepująca: do systemu wprowadzane są dwie cechy (od pojedynczego użytkownika), czyli obraz układu żył w palcu oraz odcisk palca, po czym każdy ze wspomnianych wcześniej modułów dokonuje przetworzenia odpowiedniego obrazu oraz realizuje ekstrakcję cech i tworzy na tej podstawie wektor cech. Następnie oba moduły kierują wyodrębnione wektory cech do modułu decyzyjnego. Moduł decyzyjny na podstawie swojej wiedzy (czyli zbioru danych poszczególnych użytkowników) oraz wybranej metody porównawczej podejmuje decyzję dotyczącą tożsamości danego użytkownika. W momencie, w którym obydwie cechy zwrócą informację o identyfikatorze danej osoby, realizowane jest globalne podjęcie decyzji. Globalną regułę decyzyjną możemy opisać w sposób następujący:

- Jeżeli moduły rozpoznające każdą z cech zwrócą dokładnie tę samą decyzję, wtedy staje się ona wynikiem wyjściowym systemu.
- Jeżeli moduły rozpoznające każdą z cech zwrócą różne decyzje (tzn. nie będą zgodne co do rozpoznanej tożsamości), wtedy globalną decyzją staje się ta o większej wadze.

4. Wykonane eksperymenty

Eksperymenty zrealizowane przez autorów dotyczyły trzech głównych aspektów. Pierwszym badanym elementem była skuteczność obydwu algorytmów działających indywidualnie, drugim – gdy zostaną one połączone w ramach systemu jako całości, natomiast trzecie zagadnienie było skorelowane z doborem algorytmu klasyfikującego. Musimy również nadmienić, że wszelkie badania zrealizowano z wykorzystaniem bazy danych zebranej przez autorów. Składała się ona ze 150 próbek, które pochodzą od 50 użytkowników (każdy był reprezentowany przez trzy próbki).

Omówienie rezultatów rozpoczniemy od informacji związanych z pierwszym i ostatnim z wymienionych celów badawczych. Autorzy niniejszej pracy pod uwagę wzięli zróżnicowane algorytmy uczenia maszynowego: *k*-Najbliższych Sąsiadów, *k*-Średnich, drzewa decyzyjne oraz regresję liniową. Podsumowanie tychże eksperymentów prezentujemy w ramach tabeli 1.

Najwyższą skuteczność (zarówno w przypadku układu żył w dłoni, jak i odcisku palca) zaobserwowaliśmy przy użyciu metody *k*-Najbliższych Sąsiadów oraz metodyki *leave-one-out,* która polega na usunięciu pojedynczej próbki ze zbioru danych i potraktowaniu pozostałej części bazy jako zbioru uczącego, który z kolei pozwoli nam na ocenę tożsamości wyizolowanego wektora cech. W przypadku układu żył w dłoni skuteczność naszych rozwiązań wyniosła 75%, natomiast odcisk palca pozwolił na osiągnięcie 80% prawidłowych rozpoznań tożsamości użytkownika.

TAB. 1. Podsumowanie zrealizowanych eksperymentów
TAB. 1. Summary of the conducted experiments

Cecha biometryczna	Zastosowana metoda	Uzyskany wynik
Odciski palców	*k*-Najbliższych Sąsiadów	80%
	k-Means	70%
	drzewa decyzyjne	50%
	regresja liniowa	50%
Układ żył w palcu	*k*-Najbliższych Sąsiadów	75%
	k-Means	70%
	drzewa decyzyjne	65%
	regresja liniowa	65%

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

TAB. 2. Podsumowanie eksperymentów związanych z systemem multimodalnym
TAB. 2. Summary of the experiments connected with multimodal biometrics system

Cecha	Skuteczność	Wybrana metodologia
Układ żył w palcu	75%	*k*-Najbliższych Sąsiadów
Odcisk palca	80%	*k*-Najbliższych Sąsiadów
Układ żył w palcu i odcisk palca	87%	*k*-Najbliższych Sąsiadów przy założonych priorytetach 0,6 dla odcisku palca oraz 0,4 dla układu żył w palcu

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Kolejnym testowanym zagadnieniem była skuteczność systemu w momencie, gdy zostaną połączone wszystkie analizowane cechy, czyli zostanie spełnione założenie o multimodalności systemu. Aby uniknąć sytuacji, w której obydwa moduły zwrócą informację o różnej tożsamości (co może prowadzić do braku jednoznaczności co do danych użytkownika), postanowiliśmy każdej z cech przyznać pewną niezerową

wagę, która określa jej priorytet. Najwyższy poziom skuteczności został zaobserwowany w momencie, w którym odcisk palca posiadał wagę 0,6, natomiast poziom priorytetu układu żył w dłoni był równy 0,4. Zmierzona skuteczność systemu jako całości była równa 87% (130 próbek z całego zbioru 150-elementowego zostało rozpoznanych prawidłowo). Podsumowanie zrealizowanych eksperymentów w formie prezentacji najlepszych rezultatów przedstawione zostało w tabeli 2.

Wnioskiem płynącym z wykonanych badań jest to, że przy odpowiedniej konfiguracji multimodalnego systemu biometrycznego jesteśmy w stanie uzyskać większą skuteczność rozpoznawania tożsamości człowieka po połączeniu wybranych cech biometrycznych aniżeli w przypadku każdej z nich w sposób indywidualny. Oczywiście, zgodnie z tym co zasygnalizowaliśmy wcześniej, w przypadku systemu opartego tylko i wyłącznie o dwie cechy musimy zapewnić pełną decyzyjność systemu w sytuacji, gdy obydwa moduły zwracają informację o innej tożsamości. Nasze rozwiązanie, oparte o priorytety, pozwoliło na zapewnienie pełnej sprawności funkcjonalnej proponowanego rozwiązania.

5. Podsumowanie

Podstawowym celem niniejszej pracy było zweryfikowanie, czy połączenie więcej aniżeli pojedynczej cechy biometrycznej pozwoli nam na osiągnięcie większej skuteczności niż w przypadku obydwu cech jednocześnie. Na mocy przeprowadzonych badań możemy jednoznacznie stwierdzić, że został on zrealizowany. Wykonane eksperymenty potwierdzają, że łącząc dwie cechy biometryczne, otrzymujemy wyższy poziom skuteczności aniżeli w przypadku każdej z nich oddzielnie. Wyniki wygenerowaliśmy na podstawie bazy danych, którą budowaliśmy przy pomocy własnego urządzenia akwizycji obrazu układu żył oraz skanera odcisków palców U.Are.U® 5160.

Nasze podejście będzie wciąż rozwijane. W kolejnych etapach będziemy dążyć nie tylko do zwiększenia zasobności naszej bazy danych, ale również do jej publicznego udostępnienia. Ponadto będziemy chcieli stworzyć w pełni zautomatyzowane urządzenie, które posłuży nam nie tylko jako skaner obydwu cech biometrycznych, ale również będziemy je mogli wykorzystać do zabezpieczenia zróżnicowanych obiektów.

Literatura

1. Evans D, Parish S. Predicting the First Recorded Set of Identical Fingerprint. *Journal of Interdysciplinary Science Topics*. 2015.
2. Dabouei A, Kazemi H, Iranmanesh SM, Dawson J, Nasrabadi NM. *Fingerprint Distrotion Rectification using Deep Convolutional Networks*. arXiv:1801.01198v1, 3 Jan 2018.

3. Wang R, Han C, Wu Y, Guo T. *Fingerprint Classification Based on Depth Neural Network*. arXiv:1409.5188, 18 Sep 2014.
4. Więcław Ł. A Minutiae-based Matching Algorithms in Fingerprint Recognition Systems. *Journal of Medical Informatics & Technologies*. 2009; 13: 65-71.
5. Feng J. Combining minutiaes descriptors for fingerprint matching. *Pattern Recognition*. 2008; 41: 342-352.
6. Yang G, Xi X, Yin Y. Finger Vein Recognition Based on a Personalized Best Bit Map. *MDPI Sensors*. 2012; 12: 1738-1757.
7. Tae Park G, Kim S. Hand Biometric Recognition Based on Fused Hand Geometry and Vascular Patterns. *MDPI Sensors*. 2013; 13: 2895-2910.
8. Honarpisheh Z, Faez K. An Efficient Dorsal Hand Vein Recognition Based on Firefly Algorithm. *International Journal of Electrical and Computer Engineering*. 2013; vol. 3, no. 1: 30-41.
9. Szymkowski M, Saeed K. *Finger Veins Feature Extraction Algorithm Based on Image Processing Methods*. Saeed K, Homenda W (Eds.) 17th International Conference, CISIM 2018, Olomouc, Czech Republic, September 27-29, 2018, Springer Lecture Notes in Computer Science (LNCS) Proceedings; 80-91.
10. Szymkowski M, Saeed K. *Fingerprint Feature Extraction with Artificial Neural Network*. In: Pejaś J, El Fray I, Hyla T, Eds *Advances in Soft And Hard Computing*. Springer Advances in Intelligent Systems and Computing; 86-97.
11. Siva Sundhara Raja D, Vasuki S. *Automatic Detection of Blood Vessels in Retinal Images for Diabetic Retinopathy Diagnosis*. Computational and Mathematical Methods in Medicine; 2015.
12. Zhang J, Cui Y, Jiang W, Wang L. *Blood Vessels Segmentation of Retinal Images Based on Neural Network*. 2015 ICIG 8th International Conference on Image and Graphics, Tianjin, China, Proceedings; 11-17.
13. Xu L, Luo S. A Novel Method for Blood Vessel Detection from Retinal Images. *Biomedical Engineering Online*. 2010; vol. 9, no. 14.
14. Tabędzki M, Saeed K, Szczepański A. A Modified K3M Thinning Algorithm. *International Journal of Applied Mathematics and Computational Science*. 2016; vol. 26, no. 2: 439-450.
15. Virdaus IK, Mallak A, Lee S-W. Fingerprint Verification with Crossing Number Extraction and Orientation-Based Matching. *Proceedings of The International Conference on Next Generation Computing*. 2017; 113-115.

Praca była wspierana przez grant S/WI/3/2018 przyznany przez Politechnikę Białostocką i sfinansowany ze środków Ministerstwa Nauki i Szkolnictwa Wyższego w Polsce.

Streszczenie

W ostatnim czasie coraz większym problemem staje się zapewnienie bezpieczeństwa użytkowników zróżnicowanych systemów komputerowych. Jednym z najlepszych sposobów stały się rozwiązania biometryczne. Jednakże część z nich posiada bardzo dużą wadę, jaką jest łatwośćich oszukania. W ramach niniejszej pracy proponujemy multimodalny system biometryczny, który składa się z dwóch cech: odcisku palca oraz układu żył w palcu. Ma on na celu przede wszystkim zwiększenie bezpieczeństwa użytkowników, gdyż znacznie trudniejsze jest spreparowanie dwóch cech niż jednej.

Słowa kluczowe: biometria, układ żył, systemy biometryczne, systemy multimodalne

Summary
Finger veins and fingerprint in multimodal biometrics system

Nowadays, one of the most important problem is safety of different computer systems users. One of the best solutions for this aim is biometrics security systems. However, huge amount of them have one disadvantage that is ease in spoofing. In this work we are presenting multimodal biometrics system based on two measurable traits: finger veins and fingerprint. Its main aim is to increase users security due to the fact that it is much harder to spoof two traits rather than only one.

Keywords: biometrics, veins, biometrics systems, multimodal systems

DOI: 10.24427/978-83-66391-28-4_4

Uczenie maszynowe w detekcji zmian patologicznych na obrazach okulistycznych

Maciej SZYMKOWSKI[*]
Khalid SAEED[*]
Emil SAEED[**]

1. Wstęp

Aktualnie jedną z najbardziej groźnych chorób dla organizmu ludzkiego jest cukrzyca. Na całym świecie choruje na nią ponad 425 milionów ludzi. Zgodnie z przewidywaniami światowej organizacji zdrowia (WHO) ta liczba ma zostać podwojona do 2050 roku. W wielu przypadkach zmiany cukrzycowe pojawiają się także na obrazach okulistycznych, np. obrazie dna oka czy też optycznej koherentnej tomografii (OCT). Nieleczona cukrzyca może prowadzić do nieodwracalnych zaburzeń widzenia, jak również nawet do całkowitej utraty wzroku. Bardzo często wczesne stadium zmian patologicznych jest niedostrzegalne gołym okiem nawet dla doświadczonego lekarza okulisty, jak również pacjent może nie zgłaszać pogorszenia widzenia. Wykrycie tego rodzaju zmian w ich wczesnej postaci pozwala uniknąć dalszych komplikacji, a także może zapobiec rozwojowi cukrzycy w organizmie.

Zmiany cukrzycowe widoczne na obrazie dna oka, sugerujące najczęściej konieczność wdrożenia leczenia, widoczne są w postaci wysięków twardych, czyli niewielkich odbarwień (najczęściej w kolorze żółto-czerwonym wpadającym w barwę pomarańczową). W literaturze możemy oczywiście znaleźć zróżnicowane algorytmy dotyczące detekcji tych zmian, jednakże zdecydowana większość z nich nie realizuje tejże operacji z wystarczającą dokładnością oraz przy odpowiednio krótkim czasie przetwarzania.

W ramach niniejszej pracy prezentujemy nasz autorski algorytm, który pozwala na detekcję zmian w obrazie dna oka oraz wstępne rezultaty, które zostały dzięki niemu osiągnięte. Jednym z ważniejszych etapów proponowanego rozwiązania jest klasyfikacja wykrytych punktów jako zmian patologicznych lub zmian nienoszących znamion choroby. Ten element jest realizowany z wykorzystaniem metod uczenia maszynowego. Finalnie dokonujemy kalkulacji skuteczności przygotowanego rozwiązania – jest ona szacowana na podstawie decyzji, jaką zwrócił nasz algorytm w porównaniu do decyzji doświadczonego okulisty, będącego równocześnie członkiem naszego zespołu.

[*] Politechnika Białostocka
[**] Uniwersytet Medyczny w Białymstoku

2. Aktualny stan wiedzy

W literaturze możemy odnaleźć wiele zróżnicowanych rozwiązań skorelowanych z przetwarzaniem obrazu dna oka. W ogólnym rozrachunku możemy wyróżnić dwie grupy takowych algorytmów. Pierwszą z nich są metodyki oparte o rozpoznawanie człowieka na podstawie siatkówki oka.

Interesujące rozwiązanie zostało zaproponowane w ramach pracy [1]. W tym przypadku autorzy skorzystali z sieci neuronowych w celu detekcji tożsamości człowieka. W pracy nie przedstawiono zbyt wiele detali, brakuje w niej informacji o tym, w jaki sposób nastąpiła konstrukcja sieci neuronowej oraz jakie modele sieci były brane pod uwagę w trakcie wykonywania eksperymentów. Ponadto, w rzeczonym artykule autorzy nie rozważają tego, czy system zadziała w sposób oczekiwany, gdy na siatkówce oka pojawią się jakiekolwiek zmiany patologiczne.

Rozwiązania bazujące na sztucznej inteligencji albo uczeniu maszynowym są powszechnie stosowane w przypadku rozpoznawania tożsamości człowieka na podstawie siatkówki oka. Dobrym przykładem jest [2]. W pracy tej zastosowano metody uczenia maszynowego do wyodrębnienia układu naczyń krwionośnych siatkówki oka. Autorzy wzięli również pod uwagę możliwość występowania zmian patologicznych w formie wysięków twardych. Inną pracą, która również odnosi się do segmentacji układu naczyń krwionośnych siatkówki oka w celu rozpoznawania człowieka jest [3]. W tym przypadku użyto algorytmu Harrisa do wyodrębnienia interesujących struktur. Autorzy zaproponowali także własną metodę służącą do porównania dwóch kolejnych próbek. Głównym celem prac [1-3] było jednak skupienie się na możliwości rozpoznawania człowieka, a nie detekcji zmian patologicznych.

Drugą grupą rozwiązań, która jest skorelowana z siatkówką oka, są metody obejmujące detekcję zmian patologicznych. Dominującym trendem w badaniach jest stosowanie metod sztucznej inteligencji i uczenia maszynowego do zautomatyzowanej detekcji zmian (najczęściej wysiękowych) na obrazach kolorowych dna oka. Koronnymi przykładami takich solucji mogą być [4-7]. Najbardziej popularną metodą są sztuczne sieci neuronowe [4, 5]. Algorytmy [4, 5] różnią się między sobą przede wszystkim zaimplementowanym modelem sieci oraz sposobem, w jaki wyekstrahowano niezbędne dane oraz utworzono wektor cech. W literaturze równie popularne są metody oparte o uczenie maszynowe [6, 7]. Ich główną wadą (podobnie jak w przypadku algorytmów opartych o sztuczną inteligencję) jest konieczność posiadania bardzo dużej bazy próbek, na mocy której następuje uczenie. Najczęściej proces przygotowania sieci do klasyfikacji trwa bardzo długo. W przypadku rozwiązań opartych o przetwarzanie obrazu możemy stwierdzić, że są one znacznie prostsze, jak również, że posiadają mniejszą złożoność obliczeniową.

3. Proponowane rozwiązanie

Głównym celem naszego algorytmu jest rozróżnienie obszarów posiadających zmiany patologiczne od fragmentów siatkówki, które są zdrowe. Nasze badania zostały przeprowadzone w oparciu o doświadczenia, jakie nabyliśmy w ramach poprzednich eksperymentów [8,9]. Proponowane rozwiązanie przedstawiamy w formie pseudokodu w ramach algorytmu 1. Proponowana metoda została zaimplementowana z wykorzystaniem języka programowania Java oraz matematycznego frameworku opracowanego przez autorów.

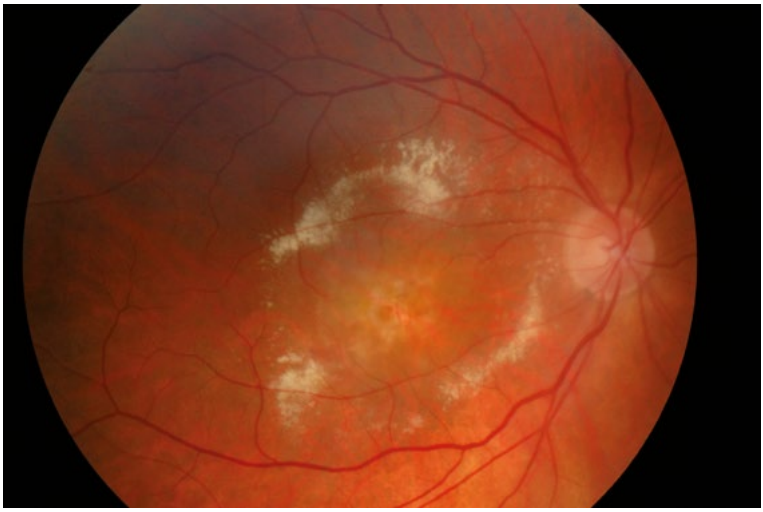
Algorytm 1. Pseudokod zaproponowanego rozwiązania

```
algorithm retina {
    image = wczytaj_obraz(ściezka)
    image.filtruj_obraz(Filtr.MEDIANOWY)
    image.konwertuj_do_skali_szarości(Kanał.ZIELONY)
    image.wyznacz_regiony()
    image.zbadaj_wyznaczone_regiony()
}
```

Pierwszym krokiem zaproponowanego rozwiązania było wczytanie obrazu. Wykonano to, wykorzystując metodę open z klasy ImageIO. Przykładowy obraz przedstawiający zdrową siatkówkę oka przedstawiamy na rysunku 1, natomiast rysunek 2 przedstawia dno oka ze zmianami patologicznymi w formie wysięków twardych.

RYS. 1. Zdjęcie przedstawiające zdrową siatkówkę oka
FIG. 1. Healthy retina sample

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 2. Zdjęcie przedstawiające siatkówkę oka z wysiękami twardymi
FIG. 2. Retina sample with hard exudates

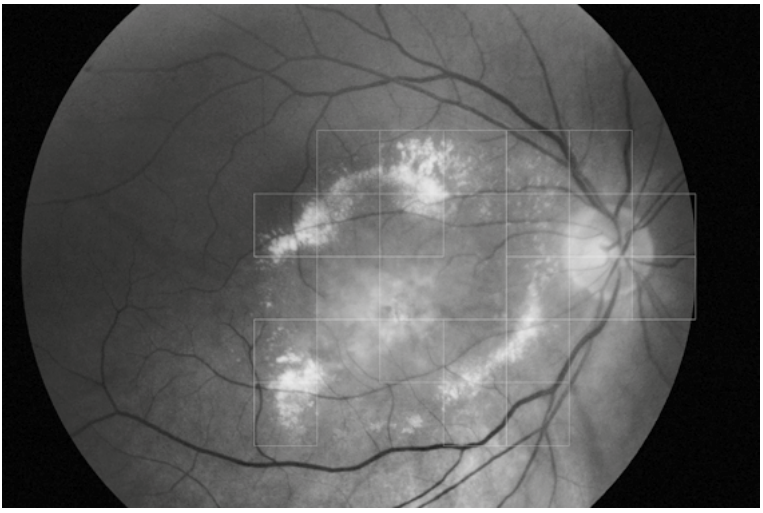
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Po wczytaniu obrazu kolejnym krokiem było usunięcie zbędnych zniekształceń, które powstają podczas procesu akwizycji danych. W tym przypadku autorzy wykonali szereg eksperymentów w celu znalezienia narzędzia, które pozwoliłoby nam na pozyskanie jak najdokładniejszego oczyszczenia obrazu. Porównywalne wyniki zostały osiągnięte przy użyciu filtru rozmywającego (przy masce 5x5) oraz filtru medianowego (z wykorzystaniem maski 3x3). Autorzy zadecydowali, że wykorzystane zostanie drugie z rzeczonych rozwiązań, ze względu na jego prostotę oraz niską złożoność obliczeniową.

Kolejnym etapem proponowanego rozwiązania była konwersja próbki do skali szarości. Podobnie jak w przypadku algorytmu układu żył w palcu, opublikowanego w [10], podjęliśmy próby dotyczące zróżnicowanych form przejścia do skali szarości. W trakcie eksperymentów zbadaliśmy konwersję z użyciem kanałów czerwonego, zielonego, niebieskiego, a także wartości uśrednionej z tych trzech elementów. Nasze badania potwierdziły tezy opublikowane w ramach prac [11, 12]. Mianowicie zaobserwowaliśmy, że najdokładniejsze odwzorowanie siatkówki oka jest obserwowalne w przypadku konwersji z użyciem kanału zielonego. W tym przypadku mamy na myśli fakt, że układ żył jest doskonale widoczny, jak również, że znacząca część dodatkowych zniekształceń została w sposób zautomatyzowany usunięta.

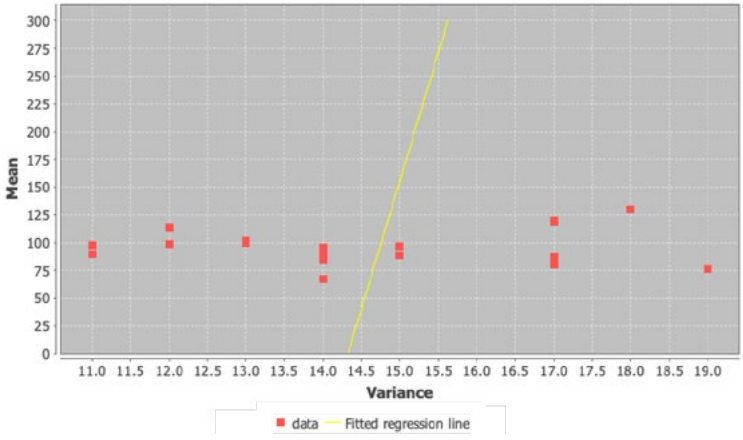
Następną procedurą użytą w ramach proponowanej solucji był algorytm skorelowany z wyznaczeniem regionów podejrzanych o bycie zmianami patologicznymi. W tym przypadku autorzy zastosowali obliczenie wartości średniej oraz wariancji. Na podstawie tych dwóch wartości (oraz dodatkowo wyznaczonych eksperymentalnie wartości progowych) podejmowana była decyzja, czy dany region może być podejrzewany

o występowanie zmian patologicznych. W trakcie badań autorzy zweryfikowali, jakiej wielkości regiony przynoszą najlepsze rezultaty. W tym przypadku okazało się, że najdokładniejszy wynik był osiągany przy podziale na sektory o wielkości 250x250 pikseli. Wynik osiągnięty po przeprowadzeniu tegoż etapu przedstawia rysunek 3.



RYS. 3. Siatkówka oka wraz z obszarami podejrzanymi o występowanie zmian patologicznych
FIG. 3. Retina with areas suspected to contain pathological changes

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 4. Przykładowa linia regresji służąca do oddzielenia obszarów zdrowych i chorych
FIG. 4. Sample regression line for separation healthy areas from the ones with pathological changes

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Ostatnim krokiem proponowanej solucji było dokładniejsze zbadanie wyznaczonych regionów. Mianowicie miało to na celu sprawdzenie, czy powinniśmy sklasyfikować region jako zdrowy czy też jako posiadający zmiany chorobowe. Operacja ta była wykonywana z wykorzystaniem metody regresji liniowej – każdy z regionów został bowiem zaprezentowany w przestrzeni dwuwymiarowej. Na jednej z osi odłożono wartość wariancji, natomiast na drugiej odchylenie standardowe. W ten sposób wyznaczono linię regresji – wartości, które znalazły się po stronie prawej uznawaliśmy za zmiany patologiczne, natomiast te po stronie lewej były klasyfikowane jako obszary zdrowe. Przykładowy wynik takiego działania został przedstawiony na rysunku 4.

4. Przeprowadzone badania

Eksperymenty, które zostały wykonane w trakcie prac nad niniejszym algorytmem możemy podzielić na trzy główne grupy. Pierwszą z nich była selekcja poszczególnych algorytmów w celu jak najdokładniejszego przygotowania obrazu do wyodrębnienia poszczególnych obszarów oraz detekcji zmian patologicznych. W tym przypadku autorzy zbadali zróżnicowane metody – m.in. filtracji, binaryzacji czy szkieletyzacji. Jednakże w poszczególnych przypadkach (jak np. po zastosowaniu metody binaryzacji) obserwowaliśmy znaczące pogorszenie osiąganych wyników. Było to nierzadko spowodowane problemem z zamazywaniem różnic pomiędzy obszarami zdrowymi a chorymi. Najlepsze wyniki (tzn. pozwalające na osiągnięcie największej dokładności) zostały osiągnięte na podstawie algorytmu opisanego w ramach trzeciej sekcji niniejszego dokumentu.

Druga grupa naszych badań dotyczyła bezpośrednio sposobu konstrukcji poszczególnych regionów. Początkowo testowaliśmy zróżnicowane kształty, poczynając od kwadratu, przechodząc przez trójkąty i prostokąty, a kończąc na okręgach. W trakcie realizowanych badań wykazaliśmy, że najdokładniejszy wynik został osiągnięty przy zastosowaniu obszarów kwadratowych. Pozostałe rozwiązania nie prowadziły do osiągnięcia aż tak dokładnych wyników, jak w przypadku rzeczonego wyboru. Kolejnym elementem związanym z obszarami była ich wielkość. Jak już zostało zaznaczone w ramach pracy, najlepsze (tzn. najdokładniejsze) wyniki uzyskano w przypadku szerokości i wysokości poszczególnych elementów równej 250px.

Ostatnim etapem badań było zweryfikowanie poziomu skuteczności proponowanego rozwiązania. W tym przypadku nie opracowaliśmy żadnej zautomatyzowanej metody, ale bazowaliśmy na decyzji naszego algorytmu, którą to porównywaliśmy do decyzji doświadczonego lekarza okulisty, będącego członkiem naszego zespołu. Na tej podstawie udało nam się ustalić, że przy próbie 150 zdjęć (na którą składało się 75 zdjęć zdrowych siatkówek oraz analogiczna ilość zdjęć ukazujących zmiany patologiczne) proponowany przez nas algorytm osiągnął skuteczność na poziomie 93,4%. W większości przypadków błędnie sklasyfikowanych obserwowaliśmy sytuację, w której zdjęcie zdrowe było rozpoznawane jako próbka prezentująca zmiany

patologiczne. Zgodnie z informacjami, jakie otrzymaliśmy od lekarza okulisty, jest to sytuacja znacznie bezpieczniejsza aniżeli przypadek, w którym próbka „chora" byłaby klasyfikowana jako „zdrowa". Znacznie lepiej jest bowiem, aby pacjent zbadał swój wzrok (w sposób dokładny i dogłębny) aniżeli zaniechał takich badań pomimo występujących (choć nie wykrytych w sposób prawidłowy) przesłanek.

5. Podsumowanie

Zrealizowane eksperymenty pokazały dobitnie, że nawet zastosowanie prostych metod statystycznych oraz podstawowych rozwiązań uczenia maszynowego może zagwarantować detekcję i klasyfikację zmian patologicznych na dosyć wysokim poziomie. Proponowany przez nas algorytm został także przetestowany na dwóch różnych urządzeniach (które to różniły się całkowicie konfiguracją i klasą posiadanych komponentów). Na podstawie tegoż eksperymentu jesteśmy w stanie także powiedzieć, że nasza solucja może być używana nawet na komputerach o niskiej jakości konfiguracji. Stanowi to zatem niewątpliwą zaletę naszego rozwiązania.

Eksperymenty w ramach naszych badań były przeprowadzone na bazie próbek wynoszącej 150 elementów, przy czym 75 z nich reprezentowało zdrowe dno oka, a kolejnych 75 było skorelowanych z siatkówką oka ze zmianami patologicznymi. Nasz algorytm pomylił się jedynie przy 10 z nich. Sposród tejże grupy 9 próbek pochodziło ze zbioru ze zmianami patologicznymi (i zostało sklasyfikowanych jako zdrowe).

Aktualnie autorzy rozpoczęli procedurę zwiększania ilości próbek w bazie oraz udoskonalania zaproponowanego rozwiązania. Ponadto kolejnym krokiem, który zostanie wykonany będzie implementacja i przetestowanie innych, bardziej złożonych algorytmów uczenia maszynowego oraz sztucznej inteligencji. Autorzy będą również dążyć do upublicznienia swojej bazy danych tak, aby inni naukowcy również mogli przeprowadzać na niej swoje eksperymenty.

Literatura

1. Sadikoglu F, Uzelatinbulat S. Biometric retina identification based on neural network. *Procedia Computer Science*. 2016; 102: 26-33.
2. Mazzaferri J, Larrivee B, Cakir B, Sapieha P, Costantino S. A machine learning approach for automated assessment of retinal vasculature in the oxygen induced retinopathy model. *Scientific Reports*. 2018; DOI: 10.1038/s41598-018-22251-7.
3. Dehghani A, Ghassabi ZR, Moghaddam HA, Moin MS. Human recognition based on retinal images and using new similarity function. *EURASIP Journal on Image and Video Processing*. 2013; 58; DOI: 10.1186/1687-5281-2013-58.

4. Anitha GJ, Maria KG. *Detecting Hard Exudates in Retinal Fundus Images Using Convolutional Neural Network*, Proceedings of International Conference on Current Trends towards Converging Technologies (ICCTCT), 2018; DOI: 10.1109/icctct.2018.8551079.
5. Bharkad S. *Morphological and Neural Network Based Approach for Detection of Exudates in Fundus Images,* 2nd International Conference on Computing Methodologies and Communication (ICCMC), 2018; DOI: 10.1109/ICCMC.2018.8487517.
6. Avula B, Chakraborty C. *Detection of Hard Exudates in Retinal Fundus Images Using Deep Learning,* 2018 7th International Conference on Informatics, Electronics and Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision and Pattern Recognition (icIVPR), Proceedings, 2018; DOI: 10.1109/ICIEV.2018.8641016.
7. Long S, Huang X, Chen Z, Pardhan S, Zgeng D. Automatic Detection of Hard Exudates in Color Retinal Images Using Dynamic Threshold and SVM Classification: Algorithm Development and Evaluation*, BioMed Research International*. 2019; 6a: 1-13.
8. Saeed E, Szymkowski M, Saeed K, Mariak Z. An Approach to Automatic Hard Exudate Detection in Retina Color Images by Telemedicine System Based on d-Eye Sensor and Image Processing Algorithms. *MDPI Sensors*. 2019; vol. 19, no. 3; DOI:10.3390/s19030695.
9. Szymkowski M, Saeed E, Saeed K, Mariak Z. *A simple algorithm for hard exudate detection in diabetic retinopathy using spectral-domain Optical Coherence Tomography,* Springer Lecture Notes in Computer Science, Proceedings of 36th International Conference on Computer Graphics, CGI 2019, Calgary, Canada, June 17-20; 179-189.
10. Szymkowski M, Saeed K. *Finger Veins Feature Extraction Algorithm Based on Image Processing Methods*. In: Saeed K, Homenda W, Eds. *17th International Conference, CISIM 2018*, Olomouc, Czech Republic, September 27-29, 2018, Springer Lecture Notes in Computer Science (LNCS) Proceedings, pp. 80-91.
11. Zhang J, Cui Y, Jiang W, Wang L. *Blood Vessels Segmentation of Retinal Images Based on Neural Network*. 2015 ICIG 8th International Conference on Image and Graphics, Tianjin, China, Proceedings; 11-17.
12. Xu L, Luo S. A Novel Method for Blood Vessel Detection from Retinal Images. *Biomedical Engineering Online*. 2010; vol. 9, no. 14.

Praca była wspierana poprzez grant S/WI/3/2018 przyznany przez Politechnikę Białostocką i sfinansowany ze środków Ministerstwa Nauki i Szkolnictwa Wyższego w Polsce.

Streszczenie

Współcześnie jedną z najbardziej groźnych chorób jest cukrzyca. W kontekście okulistyki jej wystąpienie może prowadzić do częściowej bądź całkowitej utraty wzroku. Lekarze są w stanie zaobserwować ją odpowiednio wcześnie w formie drobnych plamek na obrazie siatkówki oka. Jednakże należy podkreślić, że często zmiany w początkowym stadium mogą być niewidoczne gołym okiem. W ramach niniejszej pracy zaproponowany został algorytm do detekcji wysięków twardych na obrazach siatkówki oka z zastosowaniem metod uczenia maszynowego. Rozwiązanie to pozwoli lekarzom na wcześniejszą detekcję groźnych zmian chorobowych.

Słowa kluczowe: uczenie maszynowe, siatkówka oka, obrazy kolorowe siatkówki oka, zmiany patologiczne, przetwarzanie obrazów

Summary
Machine Learning in the detection of pathological changes in ophthalmic images

Nowadays diabetes is one of the most dangerous ilnessess. If we are thinking about our sight, untreated diabetes can lead to partial or complete vision loss. Ophthalmologists can observe early changes in retina color image in the form of small spots. However, sometimes these spots are too small to obersve them by eye. In this article we propose novel algorithm for detection of hard exudates in retina color images with machine learning methods. This solution will allow ophthalmologists early detection of dangerous pathological changes.

Keywords: Machine Learning, retina, retina color images, pathological changes, image processing

DOI: 10.24427/978-83-66391-28-4_5

Wizualizacja algorytmów wyszukiwania wzorców liniowych w obrazach 2*D*

Paweł ZABIELSKI[*]

1. Wprowadzenie

Algorytmy eksploracji danych mogą być używane m.in. w zagadnieniu wydobywania wzorców z obrazów dwuwywymiarowych (2*D*) [1]. W artykule analizowany jest problem wydobywania wzorców liniowych z obrazów 2*D*. Badane są możliwości wydobywania wzorców liniowych w obrazach z zastosowaniem minimalizacji wypukłych i odcinkowo-liniowych funkcji kryterialnych (metoda *CPL*) [2]. Minimum wypukłych i odcinkowo-liniowych funkcji kryterialnych może być efektywnie wyznaczane za pomocą algorytmu wymiany rozwiązań bazowych, który jest podobny do algorytmu *Simplex*, używanego w programowaniu liniowym.

Praca przedstawia metodę służącą do wykrywania wzorców. Jest ona nowatorskim sposobem oraz alternatywą dla transformacji Hougha, która została przedstawiona i opatentowana w 1962 roku przez Paula Hougha. Jej główne zadanie związane jest ze znajdowaniem zależności liniowych na obrazach [3]. Została ona następnie rozszerzona na ogólne klasy krzywych [4].

Idea transformacji Hougha oparta jest na fakcie, iż punkty położone współliniowo na płaszczyźnie można opisać równaniem:

$$y = ax + b \tag{1}$$

gdzie a jest nachyleniem (tangens kąta), zaś b wartością funkcji w punkcie 0. Każdy taki punkt (x, y) znajdujący się na prostej można przedstawić jako prostą w przestrzeni parametrów a i b. Wtedy punkty z prostej w przestrzeni cech przecinają się dokładnie w jednym punkcie w przestrzeni parametrów [5]. Zależność ta pozwala odnajdywać płaskie wzorce. Odpowiadające sobie punkty i proste w przestrzeni cech i parametrów zostały przedstawione odpowiednio na rysunku 1 i rysunku 2.

Reprezentacja ta ma jednak swoje minusy. Są to m.in.:

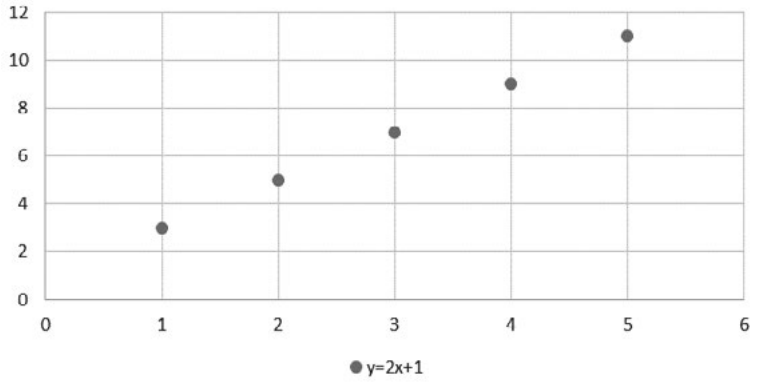
- wielkie wartości parametru a dla prawie pionowych linii,
- brak reprezentacji dla linii pionowych.

[*] Politechnika Białostocka

W celu uniknięcia powyższych problemów używa się parametryzacji w biegunowym układzie współrzędnych:

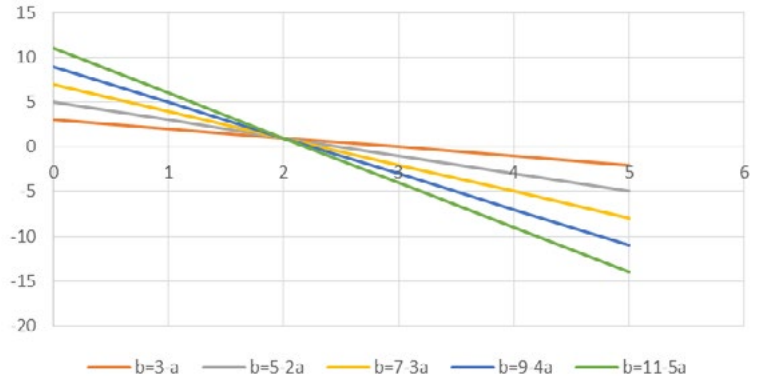
$$xcos(\theta) + ysin(\theta) = \rho \tag{2}$$

gdzie θ jest kątem nachylenia, zaś ρ odległością prostej od początku układu współrzędnych.



RYS. 1. **Przykładowe punkty w przestrzeni cech**
FIG. 1. **Examples of points in the space of features**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



RYS. 2. **Proste w przestrzeni parametrów odpowiadające punktom, które zostały przedstawione w przestrzeni cech na rysunku 1**
FIG. 2. **Lines in the parameter space corresponding to the points that were presented in the feature space in Figure 1**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Główną wadą algorytmu transformacji Hougha jest złożoność obliczeniowa oraz pamięciowa. Wraz ze wzrostem wymiarów złożoność ta rośnie wykładniczo [3].

W prezentowanym artykule zostanie przedstawiona alternatywna metoda efektywnego wykrywania płaskich wzorców, która ma zastosowanie również w przypadku zbiorów wielowymiarowych.

2. Założenia

Zbiory danych, które składają się z n wymiarowych wektorów cech $\mathbf{x}_j = [x_{j1},...,x_{jn}]^T$ ($\mathbf{x}_j \neq \mathbf{0}$), gdzie $j = 1, ..., m$, można przedstawić w n wymiarowej przestrzeni cech $F[n]$ ($\mathbf{x}_j F[n]$) jako punkty.

Płaszczyznę wierzchołkową $P_k(\mathbf{x}_{j(1)}, \ldots, \mathbf{x}_{j(l+1)})$ o wymiarze $(l - 1)$ w n-wymiarowej przestrzeni cech $F[n]$ zdefiniujmy za pomocą l liniowo niezależnych wektorów cech $\mathbf{x}_{j(i)}$, zgodnie z [6]:

$$P_k(\mathbf{x}_{j(1)}, \ldots, \mathbf{x}_{j(l)}) = \{\mathbf{x}: \mathbf{x} = \alpha_1 \mathbf{x}_{j(1)} + \ldots + \alpha_l \mathbf{x}_{j(l)}\} \tag{3}$$

Parametry α_i powinny spełniać warunek $\alpha_1 + \ldots + \alpha_l = 1$ [6].

Przedstawiona w artykule metoda ma za zadanie odkrywać punkty przecięcia poprzez odkrywanie wierzchołków zdegenerowanych – wierzchołek l-tego rzędu $\mathbf{w}_k = [w_{k,1},...,w_{k,n}]^T$ R^n jest wierzchołkiem zdegenerowanym, gdy jest punktem przecięcia więcej niż l hiperpłaszczyzn h_j:

$$(\forall j \in \{1, \ldots, m\})\ h_j = \{\mathbf{w}: (\mathbf{x}_j)^T \mathbf{w} = 1\} \tag{4}$$

Zadanie to można oprzeć na minimalizacji wypukłych i odcinkowo-liniowych (typu *CPL*) funkcji kryterialnych [6].

3. Wypukła i odcinkowo-liniowa (typu CPL) funkcja kryterialna oraz funkcja kary

Funkcja kary dla problemu wykrywania zależności kolinearnych została zdefiniowana następująco:

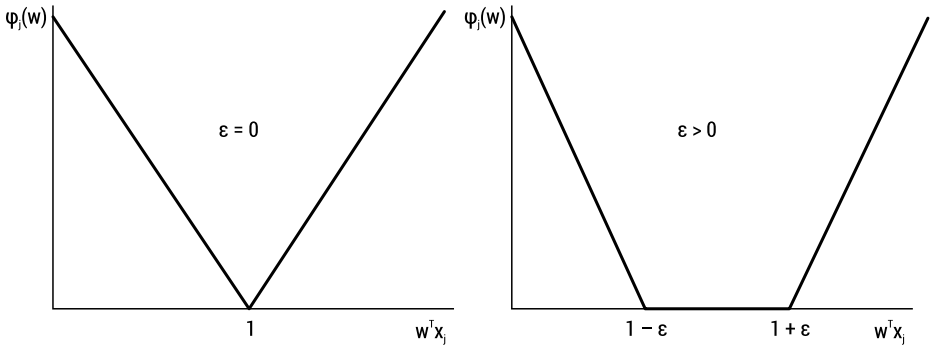
($„\mathbf{x}_j \in C_k$)

$$\varphi(\mathbf{w}) = \begin{cases} 1 - \varepsilon - \mathbf{w}^T \mathbf{x}_j & \text{dla } \mathbf{w}^T \mathbf{x}_j \leq 1 - \varepsilon \\ 0 & \text{dla } 1 - \varepsilon < \mathbf{w}^T \mathbf{x}_j < 1 - \varepsilon \\ \mathbf{w}^T \mathbf{x}_j - 1 - \varepsilon & \text{dla } \mathbf{w}^T \mathbf{x}_j \geq 1 - \varepsilon \end{cases} \tag{5}$$

ε stanowi tu margines i w zależności od niego funkcja kary wygląda jak na rysunku 3.

Funkcja kryterialna $\Phi(\mathbf{w})$ jest sumą funkcji kar (5) i przedstawić ją można w następującej postaci:

$$\Phi(\mathbf{w}) = \Sigma\, \alpha_j \varphi(\mathbf{x}_j;\mathbf{w}) \tag{6}$$



RYS. 3 Funkcja kary dla ε = 0 oraz dla ε > 0
FIG. 3 The penalty function for ε = 0 and ε > 0

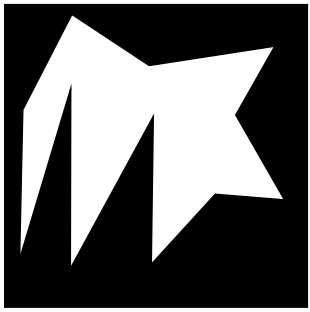
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Zastosowana implementacja dla algorytmów implementujących tranformację Hougha w optymalny sposób minimalizuje wartość funkcji kryterialnej nawet w zbiorach wielowymiarowych.

4. Wzorce liniowe w obrazach 2D

Przetwarzanie obrazów dwuwywymiarowych daje możliwość wszechstronnej wizualizacji specyfiki działania wybranych algorytmów wydobywania wzorców liniowych. Jednym z ważnych w praktyce zagadnień jest problem redukcji zakłóceń (szumów) w obrazach dwuwywymiarowych [7].

Rysunek 4 przedstawia przykładowy obraz dwuwywymiarowy. Stanowić on będzie obraz wejściowy, w którym omawiany wyżej algorytm będzie miał za zadanie wykryć wzorce liniowe.

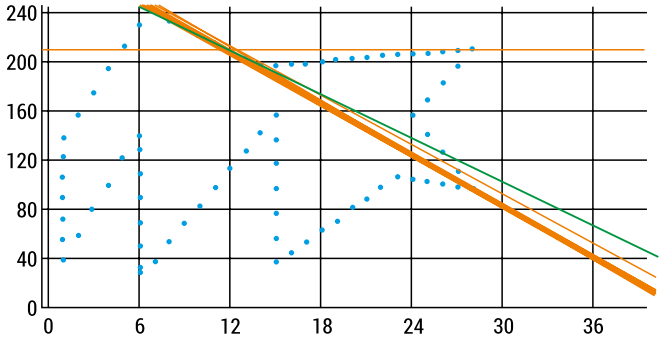


RYS. 4. Przykładowy obraz dwuwymiarowy, na podstawie którego zostanie zaprezentowane wykrywanie wzorców liniowych

FIG. 4. An example of two-dimensional image on that will be presented detecting linear patterns algorithm

ŹRÓDŁO: http://pforczmanski.zut.edu.pl/homepage/wp-content/uploads/w07-cechy_ksztaltu.pdf.
SOURCE: http://pforczmanski.zut.edu.pl/homepage/wp-content/uploads/w07-cechy_ksztaltu.pdf.

Algorytm stopniowo znajduje w każdym kroku kolejne wzorce. Rysunek 5 przedstawia kolejne kroki, które kończą się znalezieniem pierwszego rozwiązania, czyli pierwszego wzorca liniowego. Zobrazowane jest dokładnie, jak wraz z kolejnymi obiegami algorytmu zmienia się rozwiązanie. Czerwone proste pokazują dopasowywanie się wag – docelowo zielona prosta reprezentuje pierwsze znalezione rozwiązanie.

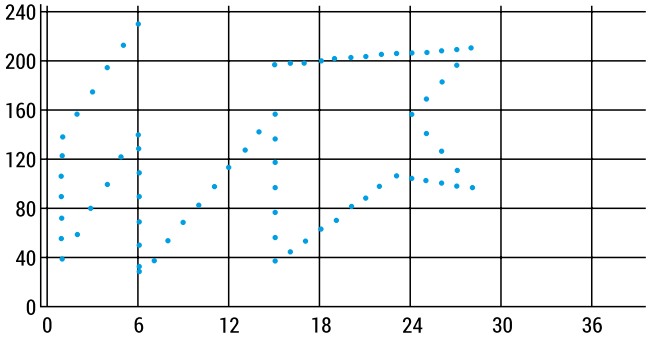


RYS. 5. Zobrazowanie kolejnych kroków algorytmu, które znajdują pierwsze rozwiązanie – pierwszy wzorzec liniowy

FIG. 5. Image of the steps of the algorithm that find the first solution – the first linear pattern

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

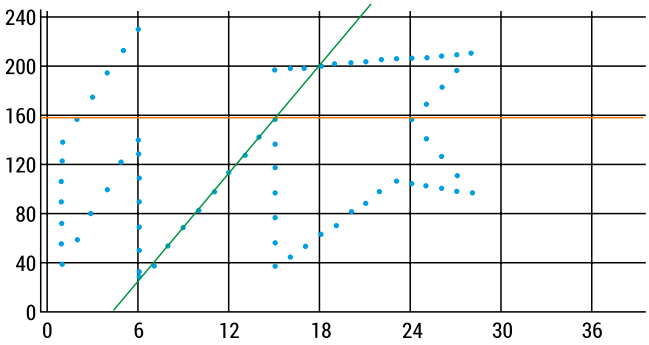
Po znalezieniu pierwszego rozwiązania usuwane są z obrazu punkty, które ono definiuje. W taki sposób do dalszych obliczeń wykorzystywany jest obraz przedstawiony na rysunku 6.



RYS. 6. **Obraz wejściowy po usunięciu punktów należących do pierwszego wzorca**
FIG. 6. **Input image after removing points belonging to the first pattern**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Kolejne kroki algorytmu mają na celu znalezienie kolejnych rozwiązań. rysunek 7 pokazuje, w jaki sposób został znaleziony drugi wzorzec. Warty uwagi jest fakt, iż drugi wzorzec zostaje znaleziony w zaledwie dwóch krokach. Przedstawia to jego szybkość – to, jak szybko jest w stanie znajdować kolejne wzorce. W podobny sposób algorytm działa do momentu, aż znajdzie wszystkie rozwiązania.



RYS. 7. **Zobrazowanie kolejnych kroków algorytmu, które znajdują drugie rozwiązanie – drugi wzorzec liniowy**
FIG. 7. **Image of the steps of the algorithm that find the second solution – the second linear pattern**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

5. Wnioski

Przedstawione opracowanie miało za zadanie zobrazować działania algorytmu minimalizacji funkcji kryterialnych typu *CPL* podczas realizacji zadania wydobywania wzorców liniowych (kolinearnych) z obrazów 2D. Algorytm ten można również dostosować do działania w zbiorach wielowymiarowych. Dzięki użyteczności w wielowymiarowych zbiorach danych algorytm ten może mieć zastosowanie m.in. w poszukiwaniu i modelowaniu interakcji na podstawie danych genetycznych. Opisana technika, korzystająca z transformaty Hougha, rozwiązuje przedstawiony problem wydobywania dwuwymiarowych wzorców kolinearnych. W dalszych pracach algorytm będzie testowany oraz optymalizowany do dzialania na wielowymiarowych zbiorach danych realnych.

Literatura

1. Duda RO, Hart PE, Stork DG. *Pattern classification*, John Wiley & Sons; 2012.
2. Bobrowski L. *Eksploracja danych oparta na wypukłych i odcinkowo-liniowych funkcjach kryterialnych*. Białystok: Wydawnictwa Politechniki Białostockiej; 2005.
3. Hough P. *Method and means for recognizing complex patterns*. U. S. Patent 3,069,654, 1962.
4. Duda OR, Hart PE. Use of the Hough Transformation to detect lines and curves in pictures. *Communications of the ACM*. 1972; 15: 11-15.
5. Illingworth J, Kittler J. A Survey of the Hough Transform, Computer Vision. *Graphics and Image Processing*. 1988; 44: 87-116.
6. Bobrowski L. *Discovering main vertexical planes in a multivariate data space by using CPL functions*. In: Perner P, Ed. *ICDM 2014*. Berlin: Springer Verlag; 2014; 200-213.
7. Bobrowski, L, Zabielski P. *Flat Patterns Extraction with Collinearity Models*, EUROSIM, Oulu; 2018; 518-524.

Streszczenie

W 1962 roku została przedstawiona i opatentowana przez Paula Hougha metoda służącą do wykrywania złożonych wzorców na zdjęciach lub innych graficznych elementach. Głównie przy zastosowaniu transformacji Hougha znajdowane były zależności liniowe. Następnie metoda ta została rozszerzona na ogólne klasy krzywych. Mimo szerokiego zastosowania i częstego użycia powszechnie znany algorytm oparty o założenia transformaty Hougha posiada ogromne zapotrzebowanie

na pamięć oraz czas wykonania. Niniejsze opracowanie przedstawia nowy algorytm wykrywania płaskich wzorców, który ma zastosowanie również w przypadku danych wielowymiarowych. Pozwala to na wykorzystanie go w przyszłości do wykrywania zależności w danych genetycznych.

Słowa kluczowe: transformata Hougha, płaskie wzorce, przestrzeń cech, przestrzeń parametrów, funkcja kryterialna

Summary
Visualization of linear patterns search algorithms in 2D images

In 1962 there was presented and patented by Paul Hough method for the detection of complex patterns in the photos or other graphic elements. Mainly using the Hough transform linear correlations were found. Then this method was extended to the general class of curves. Despite of the widespread and frequent usage of well-known algorithm based on assumptions of transformation Hough'a it has a huge demand for memory and execution time. This paper presents a new algorithm for the detection of flat patterns, which is also applicable in the case of multidimensional data. This allows to use it in the future in detection of correlation in genetic data.

Keywords: Hough'a transformation, flat patterns, feature space, parameter space, criterion function.

DOI: 10.24427/978-83-66391-28-4_6

Wybrane strategie walki z przeciwnikiem w turowej grze strategicznej

Adam DĄBROWSKI [*]
Magdalena TOPCZEWSKA [*]

1. Wprowadzenie

Gra komputerowa to oprogramowanie, którego idea wywodzi się z gier planszowych i fabularnych oraz umożliwia użytkownikowi przede wszystkim rozrywkę [6]. Często wymaga jednak także rozwiązywania skomplikowanych zadań logicznych [5].

Jednym z popularnych rodzajów gier jest strategiczna gra turowa, w której scenariusz rozgrywki podzielony jest na kilka tur. W każdej turze gracz może wykonać określone ruchy, takie jak przemieszczanie, atakowanie czy budowanie. Na uzyskane wyniki w kolejnych turach mają wpływ ruchy wykonane w turach poprzedzających, więc gracz w celu otrzymania najlepszych wyników powinien opracować strategię grania, którą będzie dostosowywał i modyfikował w zależności od ruchów przeciwnika.

Tematem niniejszego opracowania jest zaprezentowanie oraz eksperymentalne porównanie wybranych podejść strategii gry turowej w przypadku walki przeciw sobie dwóch armii. Zaprezentowane zostaną takie statystyki, jak: procent wygranych gier, procent pozostałej po rozgrywce armii, liczba potrzebnych do zakończenia rozgrywki ruchów czy optymalność wykorzystania zasobów.

2. Opis gry

Podstawą przeprowadzenia opisanych w kolejnych rozdziałach eksperymentów jest strategiczna gra turowa stworzona przez autora – Adama Dąbrowskiego – oraz Sylwię Maleszę [3, 8]. Zaprojektowano oraz stworzono świat, plansze, modele i struktury. Podstawową funkcjonalnością jest możliwość przeprowadzania potyczek pomiędzy dwoma graczami, którzy pełnią w grze rolę dowódców dwóch armii. Każdy z dowódców posiada swoje wojsko, składające się z kilku rodzajów jednostek o różnej sile. Armia podzielona jest na oddziały, a każdy oddział składa się z ustalonej liczby jednostek tego samego typu. Jednostki zostają rozlokowane na dostępnych polach planszy. Jedna figurka jednostki reprezentuje cały oddział.

Wśród jednostek armii opisywanej gry wyróżnić można (rys. 1):

- kusznika – jednostkę dystansową, czyli wykonującą ataki na odległość, słabo mobilną, posiadającą niewielką odporność na ataki fizyczne i magiczne (rys. 1a);

[*] Politechnika Białostocka

- tarczownika – jednostkę walczącą w zwarciu, czyli atakującą wyłącznie jednostki na sąsiadujących polach lub polach, do których może się przemieścić, przeciętnie mobilną, odporną na ataki fizyczne, ale nie na ataki magiczne (rys. 1b);
- maga – jednostkę dystansową, zadającą duże obrażenia magiczne, przeciętnie mobilną, z niską wytrzymałością oraz słabą odpornością na ataki fizyczne (rys. 1c);
- kawalerię – jednostkę walczącą w zwarciu, ale posiadającą duży zasięg, mającą bardzo dużą odporność na obrażenia ficzyczne i magiczne, stanowiącą najsilniejszy rodzaj jednostki w grze (rys. 1d);
- katapultę – jednostkę dystansową zadającą duże obrażenia fizyczne, bardzo mało mobilną, bardzo odporną na obrażenia fizyczne, natomiast słabo odporną na obrażenia magiczne (rys. 1e).

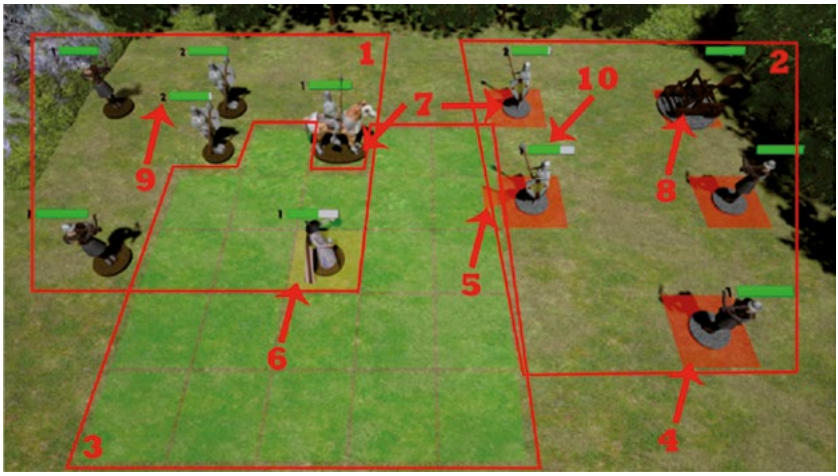
a) kusznik b) tarczownik c) mag d) kawaleria e) katapulta

RYS. 1. **Obrazy jednostek na planszy oraz ich symbole**
FIG. 1. **Images of units on the board and their symbols**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Na rysunku 1 oprócz wyglądu figur na planszy zaprezentowane zostały także ich symbole używane zarówno w samej grze, jak i w dalszej części pracy.

W trakcie przeprowadzania bitwy pomiędzy dwoma armiami na ekranie komputera widoczne są dodatkowe informacje, ułatwiające graczowi podejmowanie decyzji (rys. 2.). Są to m.in.: obydwie armie (rys. 2, zaznaczenie 1 i 2) oraz kolory ich platform (rys. 2, zaznaczenie 7); oddział, który aktualnie wykonuje ruch (rys. 2, zaznaczenie 6); podświetlenie pól, na które aktualny gracz może przemieścić swoje jednostki (rys. 2, zaznaczenie 3); pola, które mogą być zaatakowane strzałem (rys. 2, zaznaczenie 4); strzałki wskazujące obszar, który może zostać zaatakowany przez przeciwnika (rys. 2, zaznaczenie 5). Nad każdą figurą (np. rys. 2, zaznaczenie 8) przedstawiona jest dodatkowo liczba oznaczająca ilość jednostek w danym oddziale (rys. 2, zaznaczenie 9) oraz pasek życia obrazujący informację o ilości ran, jakie odniósł dany oddział (rys. 2, zaznaczenie 10).



RYS. 2. Zrzut ekranu wykonany podczas bitwy
FIG. 2. Screenshot during the battle

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Każdy gracz w trakcie swojej kolejki wykonuje jeden ruch. Pojedynczy ruch gracza polega na:

- przemieszczeniu jednostek na inne pole,
- przemieszczeniu jednostek wraz z atakiem na przeciwnika,
- ataku na przeciwnika bez przemieszczenia jednostek.

W przypadku przemieszczenia oddziałów i ataku w zwarciu można zawsze spodziewać się od przeciwnika kontrataku.

3. Algorytmy dotyczące strategii gry

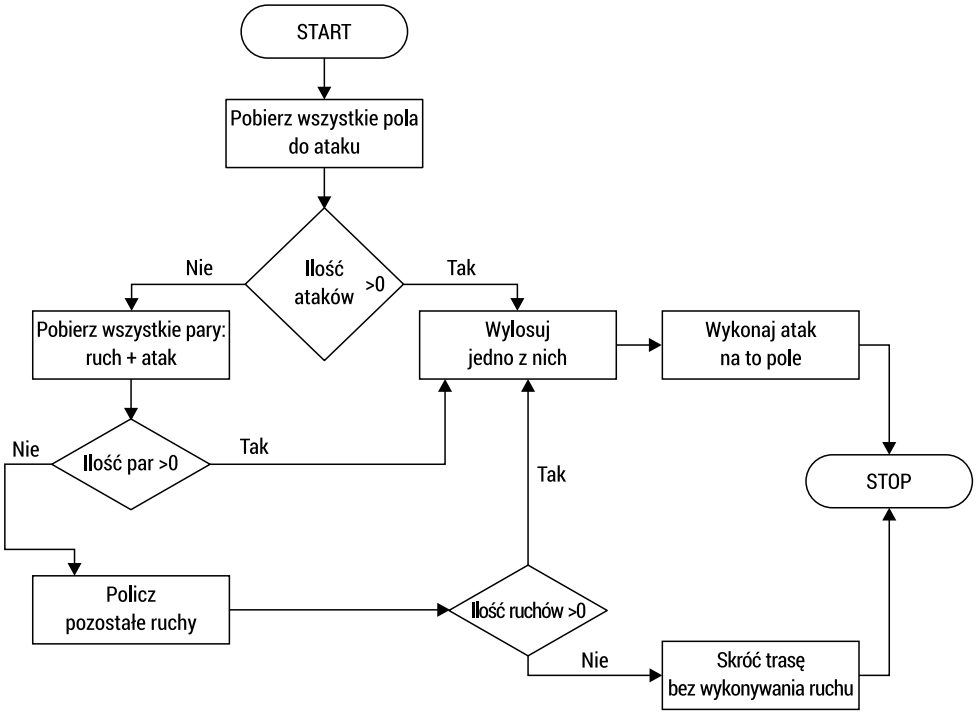
Wersja podstawowa stworzonej gry umożliwia odbycie pojedynków pomiędzy prawdziwymi zawodnikami. Istnieje także możliwość grania z komputerem, czyli z armią, która nie należy do żadnego z graczy. Aby to osiągnąć, gra została wyposażona w dodatkowy moduł odpowiedzialny za wykonywanie ruchów jednostkami.

Podstawowym problemem jest właściwy dobór ruchu spośród kilku, kilkunastu lub kilkudziesięciu możliwości. Istnieje cały szereg metod, które można wykorzystać przy implementacji sposobu poruszania się oddziałów w grze [5]. W tej pracy zaimplementowano oraz porównano trzy podejścia:

- losowe,
- najbliższego przeciwnika,
- oceny stanu na polu bitwy.

3.1. Algorytm losowy (AL)

Algorytm losowy należy do rodziny algorytmów prostych i prezentuje najprostsze podejście, będące punktem odniesienia do pozostałych algorytmów. W sposób losowy, z puli wszystkich dostępnych w danej rozgrywce ruchów, wybierany jest jeden. Schemat blokowy metody przedstawiony został na rysunku 3.



RYS. 3. Schemat blokowy algorytmu losowego
FIG. 3. Block diagram of a random algorithm

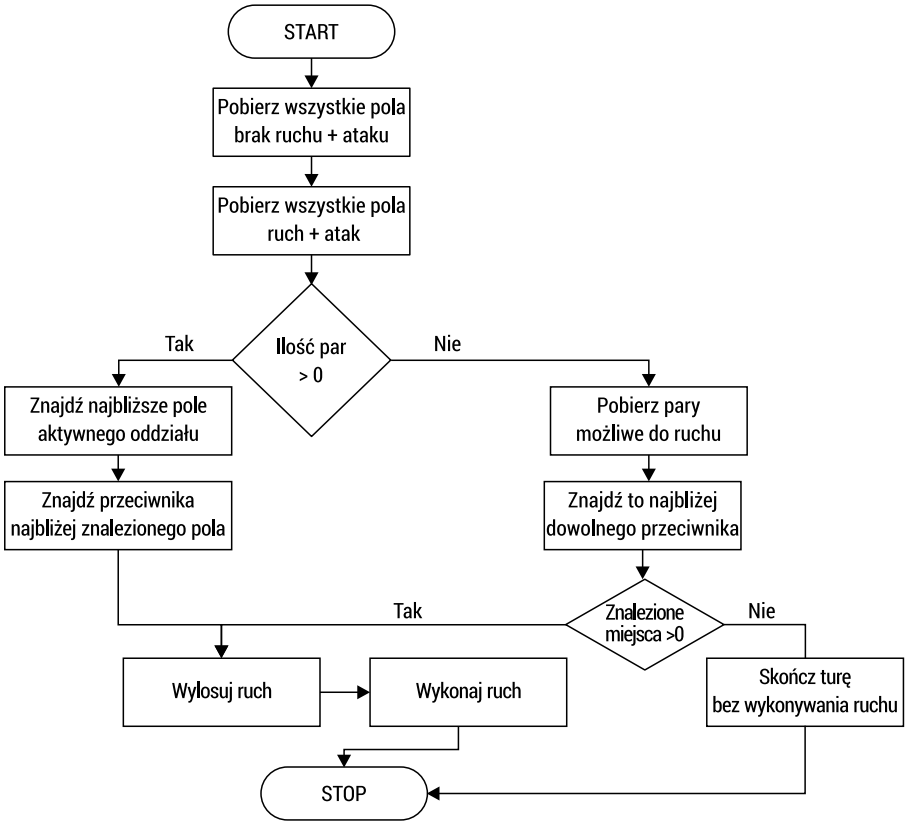
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Z przeprowadzonych eksperymentów wynika, że podejście to daje bardzo przypadkowe ruchy, w związku z czym jednostki mogą np. cofać się zamiast atakować. W konsekwencji przypadkowe działania mogą prowadzić do wydłużenia czasu trwania bitwy. W związku z tym wprowadzono modyfikację polegającą na wymuszeniu ataku, jeśli istnieje taka możliwość, eliminując opisany powyżej problem.

Dodatkowo, by uniknąć ataku odwetowego, preferowany jest atak dystansowy. W przypadku gdy w ogóle nie ma możliwości ataku, jednostki przemieszczane są w sposób losowy. Jeśli brak jest możliwości ruchu, co jest bardzo mało prawdopodobne, wówczas nie jest wykonywany żaden ruch.

3.2. Algorytm najbliższego przeciwnika (ANP)

Drugim analizowanym podejściem wyboru ruchu w grze jest atakowanie lub zbliżanie się do przeciwnika, który jest w najbliższej odległości od jednostek gracza. W każdej iteracji dla każdego oddziału obliczane są odległości euklidesowe do każdego oddziału przeciwnika, a następnie wybierany jest oddział, który znajduje się najbliżej. Dzieje się tak w przypadku, gdy konieczne jest przeprowadzenie ataku w zwarciu.



RYS. 4. Schemat blokowy algorytmu najbliższego przeciwnika
FIG. 4. Block diagram of the nearest enemy algorithm

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Istotną zaletą obydwu zaprezentowanych algorytmów jest ich prostota, która jednocześnie stanowi jednak ich ogromną wadę. Granie z przeciwnikiem wybierającym ruchy w sposób losowy lub wybierającym jedynie umiejscowione najbliżej siebie jednostki przeciwnika powoduje, że takiego gracza można pokonać w prosty sposób.

Ruchy w grze stają się przewidywalne, przez co cała rozgrywka przestaje być ciekawa dla graczy. W związku z tym zaproponowany został algorytm, który do wyboru kolejnego ruchu w grze będzie wykorzystywał ocenę aktualnej sytuacji na polu bitwy.

3.3. Algorytm oceny stanu na polu bitwy (AOSnPB)

Algorytm oceny stanu na polu bitwy wykorzystuje ideę gry w szachy, gdzie przed podjęciem decyzji o kolejnym ruchu, analizuje się dostępne możliwości, by wybrać optymalne rozwiązanie. W szachach wykorzystywany jest tzw. system oceny pozycji [1].

Do szacowania stanu na polu rozpatrywane są w grze dwa rodzaje wartości armii: materialna i niematerialna. Na wartość materialną składają się liczba oddziałów, jakie posiada gracz oraz siła jednostek, czyli ich typ. Wartość niematerialna związana jest natomiast z pozycjami, na jakich rozmieszczone są oddziały na planszy, co może mieć bezpośredni wpływ na wynik rozgrywki. Następnie, dla każdej możliwości ruchu obliczany jest i zapamiętywany stosunek wartości obu armii. Dysponując wszystkimi wynikami można ostatecznie wybrać rozwiązanie dające największą przewagę nad przeciwnikiem. Schemat blokowy algorytmu zaprezentowany został na rysunku 5.

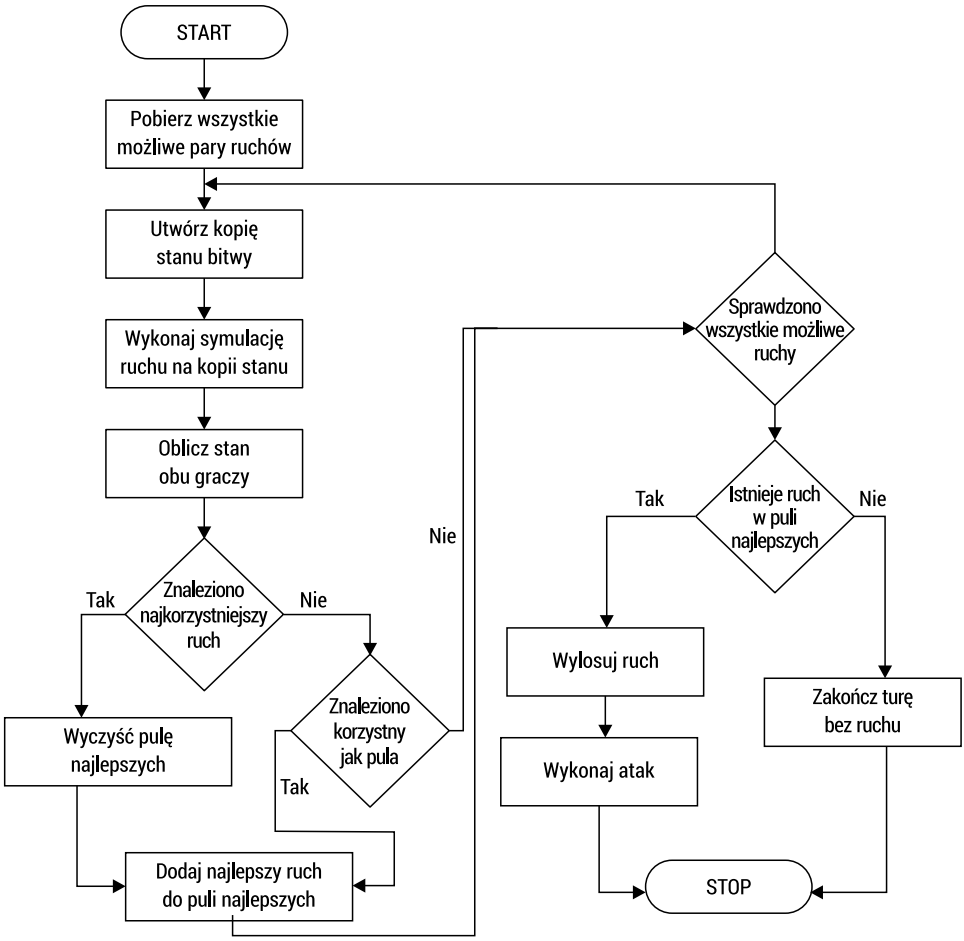
W celu stworzenia matematycznego oszacowania oceny stanu bitwy wykorzystane zostały zarówno składowa materialna, jak i niematerialna. Jednak ze względu na trudność napotkaną przy przeliczaniu wpływu liczby jednostek i ich siły na ocenę, wartość materialna została przedstawiona w postaci dwóch pochodnych składowych: siły ataku oraz wytrzymałości armii.

Ostatecznie, składowymi funkcji oceny zostało siedem czynników:

$$S = \left\{ W_a, W_w, W_k, D_z, D_d, P_p, K_r \right\} \tag{1}$$

gdzie:

- W_a – wartość ataku – wartość wpływu na obrażenia zadawane oddziałom przeciwnika; wyższa wartość oznacza większe obrażenia przeciwnika;
- W_w – wartość wytrzymałości – punkty życia każdej jednostki redukowane podczas doznawania obrażeń; im więcej punktów posiada jednostka, tym jest bardziej wytrzymała i może dłużej utrzymać się na polu walki;
- W_k – wartość kontrataku – atak obrońcy, z którym spotyka się grający po wykonaniu swego ruchu w zwarciu; niska wartość parametru powoduje brak wpływu na ocenę, a wysoka wartość może wpłynąć na wybór ruchu, w którym preferowany będzie atak i nastąpi niekorzystny dla gracza kontratak;
- D_z – odległość od przeciwnika jednostek walczących w zwarciu – jednostki walczące w zwarciu nabierają większego znaczenia w grze w przypadku, gdy ich przeciwnicy znajdują się w odległości umożliwiającej atak; mniejsza odległość między członkami własnej armii oznacza bliższą odległość od przeciwnika, a co za tym idzie – lepszy stan armii na polu bitwy;



RYS. 5. **Schemat blokowy algorytmu oceny stanu na polu bitwy**
FIG. 5. **Block diagram of the battle evaluation algorithm**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

- D_d – odległość od przeciwnika jednostek dystansowych – w przypadku jednostek dystansowych preferowana jest ich większa odległość od jednostek przeciwnika; w algorytmie nie są brane pod uwagę odległości od strzelców przeciwnika, gdyż jednostki te mogą atakować przeciwnika z każdej pozycji;
- P_p – przewaga przewagi – wpływ rozmieszczenia oddziałów i ich siły na planszy; preferowanym ustawieniem jest duża odległość od silnych oddziałów przeciwnika, a mała od słabych;
- K_r – kolejność ruchu – wpływ kolejności wyboru jednostki na możliwość ataku i szybkiego wyeliminowania z gry jednostek przeciwnika.

Poprawność implemetacji wszystkich składowych została sprawdzona eksperymentalnie poprzez przetestowanie przygotowanych scenariuszy bitew. Ustawiając wartości sześciu z siedmiu wag poszczególnych składowych na zero, sprawdzono również działanie każdej składowej osobno.

Wadą zaprezentowanego algorytmu jest konieczność wyliczenia wartości oceny dla każdej możliwości ruchu w danym momencie gry, dlatego, w porównaniu z poprzednimi, algorytm ten jest wolniejszy. Czas obliczeń jest jednak akceptowalny i niezauważalny dla gracza. Niewątpliwą zaletą jest możliwość porównania ruchów, które graczowi wydają się równie dobre oraz znajdowanie ruchów, które nie są oczywiste, a dają leszy rezultat.

Na zachowanie algorytmu można dodatkowo wpływać poprzez zmianę wag w_i funkcji oceny:

$$O = w_1 W_a + w_2 W_w + w_3 W_k + w_4 D_z + w_5 D_d + w_6 P_p + w_7 K_r \tag{2}$$

gdzie: w_i oznacza *i*-tą wagę.

Poszczególne składowe funkcji oceny (2) mogą mieć różny wpływ na ostateczny wynik ze względu na różne rzędy wielkości, dlatego pierwszą dodatkową opcją może być normalizacja składowych wyrazów. Można także wpływać na wartość wag, niwelując różnice we wpływie lub tak dobierając wartości, by wyróżniać pewne pożądane zachowania w trakcie przeprowadzania potyczki.

Do znalezienia zestawu wag można wykorzystać różne rozwiązania [4], w tym sieci neuronowe [7]. Problemem jest tu jednak jest tu brak danych – zbioru uczącego, czyli historii gier wraz z ustawieniami, na których można byłoby uczyć sieć. Potrzebne jest więc rozegranie szeregu gier, przy czym losowe ich generowanie jest zbyt kosztowne do przeprowadzenia eksperymentów. Ustalono, że w pierwszym kroku należy przetestować różne zestawy wag, zmieniając je z dużym krokiem, natomiast by wyniki były miarodajne, niezbędna jest odpowiednia liczba rozegranych gier. W celu doboru wag ograniczono się do trzech ustawień oddziałów na polu bitwy (rys. 6a, b i c) oraz przeszukiwania przestrzeni wartości w sposób opisany poniżej.

W pierwszej iteracji wartość wagi wybierana jest na podstawie informacji o tym, w jaki sposób odpowiadająca składowa ma wpływ na wynik gry, szacując, czy wartość wagi powinna być małą czy dużą liczbą. Na tej podstawie przypisuje się wagom odpowiednio wartość minimalną lub maksymalną. Drastyczne ograniczenie zbioru wartości wag powoduje, że do przetestowania otrzymano 128 kombinacji wag. Po sprawdzeniu, która z dwóch wartości wpływa na lepszy ogólny wynik, przedział wartości wagi jest następnie dzielony na dwie połowy, a do dalszych poszukiwań wybiera się połowę, w której znajduje się wartość dająca lepsze rezultaty. Na tej samej zasadzie powtarzano kolejne iteracje algorytmu poszukiwań, zawężając przedziały wartości wag. Do porównań znalezionych rozwiązań oraz wyboru najlepszego zestawu wag wykorzystany został przypadek, gdy wszystkie wagi były równe. Dla każdej kombinacji wag rozegrano 500 gier, z czego jedna rozgrywka trwała ok. 1 s. Łącznie przeprowadzono pół miliona symulacji, które trwały 120 godzin. Szczegóły doboru wag algorytmu oceny stanu na polu bitwy przedstawione zostały w [2].

4. Ekperymentalne porównanie algorytmów

Do przeprowadzenia porównań działania trzech algorytmów strategii walki z przeciwnikiem oraz oszacowania różnic w wynikach rozgrywek wykorzystano sześć ustawień oddziałów armii na planszy (rys. 6):

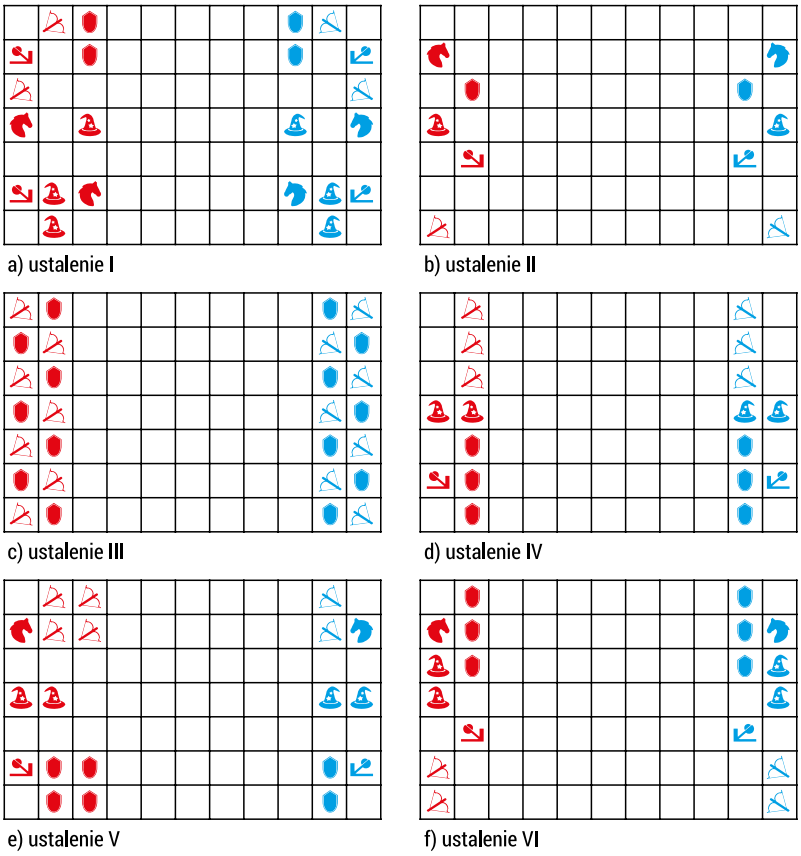
- ustawienie I – obydwaj gracze dysponują dwoma oddziałami każdego typu, rozmieszczonymi symetrycznie, choć w sposób nieuporządkowany; obecne oddziały magiczne;
- ustawienie II – obydwaj gracze dysponują pojedynczymi oddziałami każdego typu, rozmieszczonymi symetrycznie, choć w sposób nieuporządkowany; obecne oddziały magiczne;
- ustawienie III – obydwaj gracze dysponują oddziałami tylko dwóch typów: walczącymi w zwarciu i na dystans, rozmieszczonymi symetrycznie, zajmującymi dwa pełne rzędy planszy;
- ustawienie IV – obydwaj gracze dysponują oddziałami różnego typu z przewagą oddziałów o mniejszej sile (kusznicy, tarczownicy), rozmieszczonymi symetrycznie; obecne oddziały magiczne;
- ustawienie V – obydwaj gracze dysponują różnymi oddziałami różnego typu o niezrównoważonych siłach; armia gracza z lewej strony planszy ma więcej oddziałów; rozmieszczenie niesymetryczne, obecne oddziały magiczne;
- ustawienie VI – obydwaj gracze dysponują różnymi oddziałami różnego typu o niezrównoważonych siłach; armia gracza z lewej strony planszy ma więcej oddziałów – po cztery, gracza po prawej po trzy; rozmieszczenie symetryczne; obecne oddziały magiczne.

Podczas przeprowadzania potyczek gromadzone były podstawowe oraz rozszerzone statystyki gry, które zaprezentowane zostały w tabelach 1, 2 i 3 z wynikami przeprowadzonych bitew: procent wygranych gier (P_w), procent pozostałej armii (P_a), liczba ruchów (N_r) oraz optymalność ataków (O_a). Optymalność ataków O_a liczona była jako stosunek obrażeń zadanych przez oddziały do obrażeń, jakie te oddziały mogłyby zadać. Jeśli dziesięcioosobowy oddział kuszników może w jednym ataku pokonać czterech kuszników z armii przeciwnej, to w przypadku ataku na oddział dwóch kuszników wykorzystuje swój potencjał w 50%. Parametr ten w niewielu przypadkach osiąga wartość 100%. Wszystkie z zaprezentowanych statystyk są wartościami średnimi z uruchomionych 500 gier dla każdego typu eksperymentu.

Eksperyment 1

Eksperyment pierwszy opisuje przypadki ustawień, w których obydwie armie mają równoważne siły. Dotyczy to ustawień: I, II, III i IV. W tabeli 1 znajdują się uzyskane wynikowe statystyki przeprowadzonych eksperymentów i symulacji walk. Prezentowane są wyniki symulacji dla armii wykorzystującej do wyboru ruchu algorytm

najbliższego przeciwnika przeciw armii wykorzystującej do wyboru ruchu algorytm losowy, wyniki symulacji dla armii wykorzystującej algorytm oceniający stan na polu bitwy przeciw armii stosującej algorytm losowy oraz wyniki bezpośredniej walki, gdy używane są algorytmy ANP oraz AOSnPB.



RYS. 6. **Wybrane ustawienia oddziałów**
FIG. 6. **Selected army units arrangements**
ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Analizując wyniki uzyskane w przypadku ustawienia I (tab. 1), można stwierdzić, że najlepsze rezultaty uzyskano dla algorytmu oceniającego stan na polu bitwy: procent wygranych gier był na poziomie 90,40% w walce wykorzystującej przez przeciwnika algorytm losowy (76,40%), a średni procent zachowanej armii wyniósł 32,57% w porównaniu z 21,12% dla algorytmu losowego. Również optymalność wykorzystania zasobów była o 1.79% wyższa w stosunku do algorytmu losowego. W bezpośrednim starciu większy procent wygranych uzyskał natomiast algorytm AOSnPB: 73,40% przy 20,90% zachowanej armii.

TAB. 1. Wyniki przeprowadzonych potyczek przy czterech wybranych ustawieniach
TAB. 1. Results of the battles for four selected army units arrangements

Ustawienie	Walka		P_w [%]	P_a [%]	N_r	O_a [%]
I	przeciw AL	ANP	76,40	21,12	18,46	67,90
		AOSnPB	90,40	32,57	18,22	69,69
	bezpośrednia	ANP	26,60	4,22	12,75	72,34
		AOSnPB	73,40	20,90	20,48	68,74
II	przeciw AL	ANP	0,00	0,00	113,47	69,82
		AOSnPB	100,00	47,03	139,22	68,05
	bezpośrednia	ANP	0,00	0,00	80,08	77,28
		AOSnPB	100,00	50,13	134,29	68,23
III	przeciw AL	ANP	76,40	16,10	49,91	68,64
		AOSnPB	96,60	29,25	46,93	71,83
	bezpośrednia	ANP	3,40	0,27	32,35	72,76
		AOSnPB	96,60	29,42	48,03	70,82
IV	przeciw AL	ANP	54,60	10,12	52,43	71,55
		AOSnPB	95,40	30,67	49,86	72,68
	bezpośrednia	ANP	13,20	1,00	40,28	79,22
		AOSnPB	86,80	28,28	50,76	71,58

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Symulacje bitew w ustawieniu II dostarczyły ciekawych rezultatów. W przypadku algorytmu AOSnPB ponownie otrzymano najwyższe wyniki: wszystkie gry zostały wygrane przy jego użyciu przy zachowaniu prawie 50% oddziałów zarówno w starciu z armiami używającymi algorytmu losowego, jak i w bezpośredniej walce. Zaskakującym wynikiem przy tym ustawieniu były wszystkie przegrane przez armie wykorzystujące algorytm ANP oraz ich całkowita utrata oddziałów. Zauważyć należy, że pomimo przegranych algorytm ANP charakteryzował się lepszą optymalizacją

wykorzystanych zasobów niż algorytm AOSnPB. Taki przypadek mógł być spowodowany faktem, że najbliższymi przeciwnikami były bardzo wytrzymałe oddziały, których nie udało się pokonać ze względu na zbyt małą siłę atakujących.

Ustawienie III jest przykładem typowego ustawienia jednostek w grze strategicznej. Tym razem również wykazano, że w przypadku algorytmu AOSnPB uzyskano lepsze wyniki. Różnice pomiędzy algorytmem ANP oraz AOSnPB w grze przeciw ruchom wybieranym przez algorytm losowy wyniosły 20,2% w średniej liczbie wygranych gier na korzyść algorytmu AOSnPB. W starciu bezpośrednim algorytm ANP wygrał jedynie w 3,40% przeprowadzonych bitew.

Kolejnym typowym przykładem ustawienia w grze strategicznej jest ustawienie IV. Różnica w procencie wygranych gier algorytmu AOSnPB w porównaniu do ANP wyniosła 40.8%. Procent zachowanej armii również różnił się na korzyść AOSnPB o 20.55%, przy konieczności wykonania średnio ok. 50 ruchów przez oba algorytmy. Różnica w optymalności ataków pozostawała na poziomie 1%. Jeszcze większe różnice w wynikach można zauważyć w przypadku bezpośrednich starć armii wykorzystujących algorytmy ANP oraz AOSnPB.

Eksperyment 2

Eksperyment drugi dotyczy walki z przeciwnikiem w sytuacji, gdy armie nie mają równoważnych sił. Dotyczy to ustawień V i VI. Jeśli przeciwnik dysponuje słabszą armią, to celem w walce może być nie tylko być wygrana, ale równoważnym celem może stać się odniesienie jak najmniejszych strat w armii. Jeżeli przeciwnik dysponuje silniejszą armią, poza próbą wygrania mimo wszystko, celem może stać się spowodowanie jak największych strat w armii przeciwnej. Aspekty te również były również badane w pracy. Wyniki zostały zaprezentowane w dwóch tabelach. W tabeli 2 znajdują się wyniki porównania algorytmów ANP przeciw AL oraz AOSnPB przeciw AL, natomiast w tabeli 3 umieszczone zostały wyniki bezpośredniego porównania walk armii wybierającej ruch na podstawie ANP oraz AOSnPB.

Analizując siły armii w ustawieniu V, gracz, którego armia rozmieszczona jest po lewej stronie planszy, ma około 30% więcej sił – głównie dzięki dysponowaniu większą liczbą oddziałów. Ustawienie VI jest ustawieniem symetrycznym, zaś różnica polega na tym, że gracz z armią po lewej stronie planszy ma każdego oddziału o jeden więcej w porównaniu z przeciwnikiem.

W przypadku rozgrywek ze słabszym przeciwnikiem (tab. 2), wykorzystującym algorytm losowy do wyboru ruchu, oba algorytmy (ANP, AOSnPB) wygrały wszystkie bitwy, co nie jest zaskakujące. Dużą różnicę można natomiast zobaczyć w stanie armii po walce: dla algorytmu AOSnPB procent zachowanej armii przy ustawieniu V wyniósł 63,24% i był o około 9% większy niż dla algorytmu ANP. Różnica jest jeszcze większa przy ustawieniu VI i wynosi około 20%. Algorytm AOSnPB pokonał ponad trzy razy więcej armii przeciwnika niż stracił. Ponadto wykorzystanie algorytmu AOSnPB umożliwiło pokonanie przeciwnika przy wykonaniu mniejszej liczby ruchów oraz optymalne wykorzystanie zasobów.

TAB. 2. Wyniki przeprowadzonych potyczek przy dwóch wybranych ustawieniach przeciw algorytmowi losowemu AL

TAB. 2. Results of the battles for six selected army units arrangements against AL random algorithm

	Ustawienie		P_w [%]	P_a [%]	N_r	O_a [%]
Słabszy przeciwnik	V	ANP	100,00	54,35	54,39	70,00
		AOSnPB	100,00	63,24	48,37	72,39
	VI	ANP	100,00	58,96	42,61	75,10
		AOSnPB	100,00	78,95	33,75	79,83
Silniejszy przeciwnik	V	ANP	0,20	0,03	27,57	73,49
		AOSnPB	33,60	4,57	37,09	74,27
	VI	ANP	0,02	0,02	27,82	90,22
		AOSnPB	14,80	2,33	50,57	80,27

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Analizując wyniki przy rozgrywkach z silniejszym przeciwnikiem (tab. 2), zamieniono armie stronami, czyli przeciwnik dysponował liczniejszą o około 30% armią. Należy zwrócić uwagę na to, że pomimo dużej dysproporcji w siłach armii wykorzystanie algorytmu AOSnPB do wybierania ruchu pozwoliło na wygranie 14,80% w starciu z silniejszym przeciwnikiem. W przypadku algorytmu ANP było to jedynie 0,02% przypadków. Ciekawym zjawiskiem jest znacznie, bo prawie dwukrotnie, większa liczba ruchów potrzebnych AOSnPB w rozgrywce w porównaniu z algorytmem ANP oraz niższa o około 10% optymalność wykorzystania zasobów, wskazująca na zastosowanie zupełniej innej strategii walki w przypadku obu algorytmów.

Wyniki walki bezpośredniej armii wykorzystujących dwa algorytmy ANP oraz AOSnPB przedstawione są w tabeli 3. W przypadku przewagi armii wykorzystującej algorytm AOSnPB wszystkie bitwy zostały wygrane w obydwu analizowanych ustawieniach. Nie udało się potwierdzić tego wyniku dla algorytmu ANP: w ustawieniu V wygranych zostało 69% starć, a w VI 93,4%. Algorytm AOSnPB uzyskał w ustawieniu V 31% wygranych starć.

TAB. 3. Wyniki przeprowadzonych potyczek przy dwóch wybranych ustawieniach
TAB. 3. Results of the battles for six selected army units arrangements

	Ustawienie		P_w [%]	P_a [%]	N_r	O_a [%]
Przewaga armii ANP	V	ANP	69,00	12,47	60,64	72,72
		AOSnPB	31,00	4,53	40,70	74,79
	VI	ANP	93,40	30,39	45,27	76,39
		AOSnPB	6,60	0,96	53,76	83,52
Przewaga armii AOSnPB	V	ANP	0,00	32,04	18,65	75,83
		AOSnPB	100,00	60,83	45,38	72,36
	VI	ANP	0,00	0,00	20,62	91,84
		AOSnPB	100,00	77,35	33,27	80,55

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Ciekawym spostrzeżeniem jest to, że w ustawieniu V armia wykorzystująca algorytm AOSnPB wygrała około pięć razy więcej gier niż w ustawieniu VI, co mogłoby oznaczać, że lepsze działanie algorytmu AOSnPB związane jest z preferowaniem walk z większą liczbą jednostek przeciwnika.

5. Wnioski

W pracy porównane zostały trzy algorytmy strategii walki z przeciwnikiem; algorytm losowy, najbliższego przeciwnika oraz oceniający stan na polu bitwy. Wykazano przewagę algorytmu AOSnPB w porównaniu z pozostałymi dwoma algorytmami oraz oszacowano wartość tej przewagi na podstawie takich statystyk jak procent wygranych gier, liczba ruchów, ale również oceniając stan armii: procent zachowanej armii oraz optymalność wykorzystania zasobów. Na podstawie eksperymentów wykazano także przewagę algorytmu ANP nad algorytmem losowym.

Zaprezentowany algorytm oceny stanu na polu bitwy może być dalej rozwijany w kilku możliwych kierunkach. Pierwszym z nich jest analizowanie nie jednego, ale całego ciągu ruchów gracza. Wymaga to jednak dużych zasobów obliczeniowych. Innym pomysłem jest dalsza praca nad algorytmem dobierania wag w funkcji oceny stanu na polu walki w celu znalezienia rozwiązania optymalnego.

Literatura

1. Botwinnik M. *Computers in Chess*. Springer-Verlag; 1984.
2. Dąbrowski A. *Porównanie strategii walki z przeciwnikiem na polu bitwy w turowej grze strategicznej*, praca magisterska; 2018.
3. Dąbrowski A, Malesza S. *Strategiczna gra rozgrywana w systemie tur z elementami RPG*, praca inżynierska; 2017.
4. Goldberg DE, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publ. Comp., Inc.; 1989.
5. Pawlewicz J. *Techniki sztucznej inteligencji w programach grających*; 2010. Dostępny w Internecie: https://www.mimuw.edu.pl/~pan/gry.pdf.
6. *Słownik Gracza*. Dostępny w Internecie: https://www.gry-online.pl.
7. Tadeusiewicz R. *Sieci neuronowe*. Warszawa: Akademicka Oficyna Wydawnicza RM; 1993.
8. *Unreal Engine 4 Documentation*. Dostępny w Internecie: https://docs.unrealengine.com/en-us.

Streszczenie

W pracy przedstawiono porównanie wybranych trzech algorytmów wyboru ruchu w trakcie rozgrywki w strategicznej grze turowej: algorytm losowy (AL), algorytm najbliższego przeciwnika (ANP) oraz algorytm oceniający stan na polu bitwy (AOSnPB). W celu przeprowadzenia eksperymentów wybrano sześć ustawień obu armii na planszy, przy czym cztery pierwsze ustawienia dotyczyły przypadku zrównoważonych sił w obu armiach, natomiast dwa kolejne dotyczyły gry z silniejszym i ze słabszym przeciwnikiem. Algorytmy ANP oraz AOSnPB porównywano z algorytmem losowym jako punktem odniesienia oraz pomiędzy sobą, by oszacować zarówno prawdopodobieństwo wygranej, jak i bardziej szczegółowe statystyki, np. stan armii po bitwie. Jest to szczególnie ważne w przypadku braku zrównoważenia sił po obu stronach. Wykazano, że algorytm oceniający stan na polu bitwy daje znacząco lepsze wyniki w porównaniu z pozostałymi algorytmami oraz oszacowano te różnice.

Słowa kluczowe: gra turowa, gra strategiczna, algorytm

Summary
Selected battles strategies in turn-based strategy game

The paper presents a comparison of three selected algorithms of motion selection during the game in a turn-based strategic game: random algorithm (AL), nearest opponent (ANP) and assessing the state on the battlefield (AOSnPB). In order to conduct experiments, six settings of both armies were chosen on the board, the first four concerned the case of balanced forces in both armies, while the two concerned games with a stronger and a weaker opponent. The ANP and AOSnPB algorithms were compared with a random algorithm as a reference and among themselves to estimate both the probability of winning and more detailed statistics, such as the state of the army after

the battle. This is especially important if the balance of forces on both sides is violated. It has been shown that the algorithm assessing the state on the battlefield gives significantly better results compared to other algorithms, and the differences were estimated.

Keywords: turn-based game, strategy game, algorithm

DOI: 10.24427/978-83-66391-28-4_7

Kompakcja wejściowego strumienia bitowego z użyciem metod zbiorów przybliżonych do symulacji mocy w układach sekwencyjnych

Tomasz GRZEŚ[*]

1. Wprowadzenie

Minimalizacja mocy oraz projektowanie układów o obniżonym poborze energii jest jednym z wiodących trendów we współczesnej technice cyfrowej. Minimalizacja zużycia energii prowadzi m.in. do wydłużenia czasu między cyklami ładowania baterii w systemach mobilnych. Służy również do projektowania urządzeń przyjaznych środowisku. Projektowanie układów o obniżonym poborze mocy i projektowanie zorientowane na moc w jednym z etapów wymagają obliczenia mocy pobieranej przez system. Jedną z możliwości jest symulacja układu w celu wyznaczenia poboru mocy, ale proces ten może być bardzo czasochłonny, ponieważ wymaga długiego wejściowego strumienia bitowego, który jest przekazywany do wejść symulowanego obwodu. Aby skrócić czas symulacji strumień można poddać procesowi kompakcji, tzn. zmniejszenia długości.

Teoria zbiorów przybliżonych została opracowana przez prof. Zdzisława Pawlaka w latach 80. XX wieku. Zbiory przybliżone są używane do analizy i przetwarzania danych, a jednym z pól ich wykorzystania jest redukcja danych. Usunięcia nadmiarowości w wejściowym strumieniu bitowym można dokonać wykorzystując metody zbiorów przybliżonych; z pomocą mogą przyjść także redukcja (redukt) i dyskretyzacja [15].

W przypadku opisanym w niniejszym opracowaniu do kompakcji wejściowego strumienia bitowego wykorzystywany jest redukt. Proponowany algorytm potrzebuje dwóch informacji: wejściowego strumienia bitowego i listy przejść układu sekwencyjnego. Strumień danych jest traktowany jako tabela decyzyjna rozszerzona o dane wyodrębnione z listy przejść w celu utworzenia atrybutu decyzyjnego. Tak przygotowana tablica jest redukowana, a następnie poddawana dyskretyzacji, co pozwala na zmniejszenie zarówno ilości bitów, jak i ilości słów w strumieniu.

Wcześniejsze prace związane z kompakcją wejściowego strumienia bitowego wykorzystywały metody wymagające skomplikowanych obliczeń arytmetycznych (np. transformata Fouriera), przez co czas ich wykoania znacznie się wydłużał. Zastosowanie

[*] Politechnika Białostocka

metod zbiorów przybliżonych pozwala na realizację algorytmu, wykorzystując prawie wyłącznie operacje boolowskie, które można np. zrealizować w układzie FPGA [5]. Prowadzi to do znaczącego przyspieszenia operacji.

Niniejsze opracowanie jest zorganizowane następująco: w punkcie drugim opisane zostały zagadnienia związane z mocą w układach cyfrowych, trzeci prezentuje techniki związane z wyznaczaniem mocy w układach sekwencyjnych oraz opisuje metody kompakcji strumienia wejściowego, czwarty przedstawia zagadnienia związane ze zbiorami przybliżonymi, w piątym został opisany proponowany algorytm oraz wyniki jego badań, natomiast w szóstym następuje podsumowanie.

2. Moc w układach cyfrowych

Pobór mocy przez układy CMOS jest spowodowany następującymi czynnikami:

- prądem upływności zależnym od fizycznych parametrów układu, takich jak technologia i sposób wykonania, wymiary elementów, jakość materiałów itp.;
- przepływem prądu związanego z przełączaniem (ang. *crowbar current*), gdy prąd płynie podczas przełączania z VDD do masy;
- przepływem prądu, który powoduje ładowanie lub rozładowanie pojemności wyjściowych każdego z elementów [11].

Należy jednak zaznaczyć, że większość mocy w układach CMOS jest wydzielana głównie na pojemnościach wyjściowych.

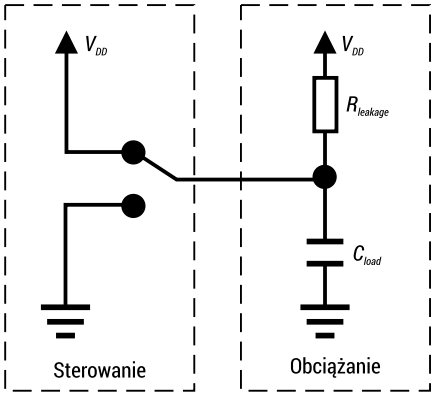
Przed rozpoczęciem obliczania mocy układów logicznych CMOS należy założyć, że [17]:

- moc jest tracona wyłącznie w pojemnościach znajdujących się na wyjściach elementów układu;
- prąd płynie albo od V_{DD} do pojemności wyjściowej albo od pojemności wyjściowej do masy;
- napięcie na wyjściu elementu logicznego (bramki, przerzutnika) może się zmieniać jedynie z V_{DD} na 0 lub z 0 na V_{DD}.

W związku z tym moc *Pa* pobieraną przez element układu cyfrowego (logicznego) *a* można wyliczyć korzystając z równania [17]:

$$P_a = \frac{1}{2} \cdot V_{DD}^2 \cdot f_{max} \cdot N_a \cdot C_a \tag{1}$$

gdzie: V_{DD} – napięcie zasilające układ; f_{max} – maksymalna częstotliwość pracy układu (maksymalna częstotliwość, względem której wyliczane są parametry sygnałów wejściowych i wyjściowych); C_a – pojemność wyjściowa elementu *a*; N_a – aktywność przełączania (średnia liczba zmian stanu wyjścia w czasie trwania jednego cyklu zegara) elementu *a*.



RYS. 1. Uproszczony schemat inwertera CMOS (na podstawie [12])
FIG. 1. Simplified schematic of the CMOS inwerter (based on [12]).

ŹRÓDŁO: [12]
SOURCE: [12].

CMOS jest technologią najczęściej wykorzystywaną w produkcji układów cyfrowych. W układach CMOS moc jest wydzielana głównie na pojemnościach wyjściowych (C_{load}) oraz na rezystancjach odpowiadających za upływność ($R_{leakage}$). Najczęściej korzysta się z modelu inwertera przedstawionego na rysunku 1. W modelu układ sterowania dołącza do obciążenia (wyjścia) bramki napięcie zasilania V_{DD} lub masę – w zależności od wartości, która ma zostać osiągnięta na wyjściu. Każdorazowa zmiana stanu wyjścia powoduje przepływ prądu związany z przeładowaniem (ładowaniem lub rozładowaniem) pojemności obciążenia C_{load}, a co za tym idzie – wydzielenie mocy.

Równanie (1) pozwala określić moc pobieraną przez element układu. Wszystkie parametry występujące po jego prawej stronie, za wyjątkiem N_a, są wartościami stałymi, zależnymi od technologii wykonania układu scalonego. W związku z tym problem obliczania mocy można zredukować do problemu obliczania aktywności przełączania układu.

3. Metody obliczania mocy w układach cyfrowych

Obliczanie mocy w układach cyfrowych jest jednym z ważniejszych procesów w projektowaniu systemów cyfrowych. Jest niezbędne do przeprowadzenia minimalizacji poboru mocy, jak również do oceny finalnego rozwiązania pod kątem poboru energii. Technicznie wyróżnia się dwie grupy metod obliczania mocy: metody statyczne oraz metody dynamiczne [3].

Statyczne metody obliczania mocy opierają się na własnościach statystycznych (probabilistycznych) badanego układu. Do obliczeń wymagana jest znajomość takich parametrów sygnału wejściowego, jak prawdopodobieństwo pojawienia się jedynki na linii wejściowej czy współczynniki korelacji pomiędzy sygnałami dla poszczególnych linii wejściowych [4].

W przypadku układów kombinacyjnych szacowanie mocy polega na wyliczeniu prawdopodobieństwa zmiany stanu dla wyjść wszystkich bramek w układzie. Aby wyliczyć prawdopodobieństwo zmiany stanu należy wyliczyć prawdopodobieństwo, że przedstawiona poniżej funkcja *f* przyjmie wartość 1 [2]:

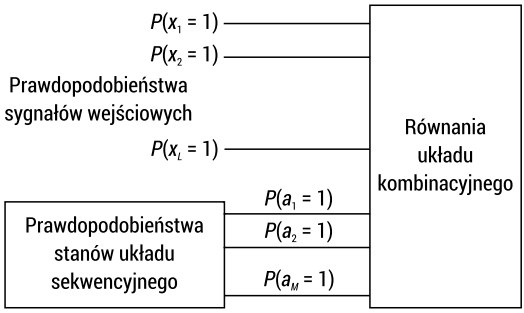
$$f = y_{t-1} \cdot \overline{y_t} + \overline{y_{t-1}} \cdot y_t \tag{2}$$

gdzie: y_t – stan wyjścia w czasie *t*; y_{t-1} – stan wyjścia w czasie $t - 1$. Gdy funkcja *f* przyjmie wartość 1, wtedy nastąpi zmiana wartości na wyjściu *y*.

W przypadku układów sekwencyjnych stosuje się modelowanie za pomocą dyskretnych łańcuchów Markowa $\left\{X^t \mid t \in T\right\}$ ze skończoną liczbą stanów $A = \{a_1, ..., a_M\}$ przy założeniu, że przestrzeń czasu *T* jest dyskretna [16]. Do obliczenia prawdopodobieństw statycznych wykorzystuje się natomiast równania Chapmana-Kołmogorowa.

Wyliczone prawdopodobieństwa statyczne pozwalają na obliczenie mocy wydzielonej w pamięci układu sekwencyjnego (czyli w przerzutnikach). Obliczenie mocy całkowitej wymaga dodatkowo obliczenia mocy wydzielonej w części kombinacyjnej układu.

Na rysunku 2 przedstawiono graficznie sposób obliczania mocy w części kombinacyjnej układu sekwencyjnego. Jako prawdopodobieństwa sygnałów wejściowych układu kombinacyjnego są podawane zarówno wartości dla wejść układu sekwencyjnego, jak i prawdopodobieństwa poszczególnych stanów układu sekwencyjnego. Na tej podstawie wylicza się aktywność przełączania, a zarazem moc wydzieloną w części kombinacyjnej układu.



RYS. 2. Statyczna metoda obliczania mocy w układach sekwencyjnych
FIG. 2. Static method of power calculation in sequential circuits

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Dynamiczne metody obliczania mocy oparte są na symulacji. Układ pobudzany jest stworzonym wcześniej ciągiem wejściowym. Ciąg ten może być budowany na podstawie parametrów statystycznych, bądź wynikać z przeznaczenia układu, np. w przypadku układu kodującego można podać ciąg bajtów, który ma zostać zakodowany.

Symulacja może być przeprowadzona na dwa sposoby. W pierwszym przypadku na wejście układu podawany jest cały wygenerowany ciąg wejściowy. Wymaga to przeprowadzenia pełnego cyklu obliczeń. Można również przeprowadzić symulację dla niepełnego ciągu – wówczas obliczenia są przerywane, gdy średnia moc zmienia się tylko w zadanym zakresie (następuje konwergencja).

Do przeprowadzenia symulacji układu potrzebne jest wygenerowanie ciągu wejściowego. Ciąg może być tworzony na podstawie danych statystycznych opisujących prawdopodobieństwa pojawienia się sygnałów wejściowych i korelacje pomiędzy nimi. Można również zastosować rzeczywisty ciąg. W celu zminimalizowania czasu symulacji przygotowywany ciąg powinien być jak najkrótszy, a jednocześnie nie powinien znacząco ujemnie wpływać na wynik symulacji (nie powinien powodować powstawania błędu).

W pracy [10] przedstawiono jedną z metod generowania ciągu wejściowego. Zastosowano tu analizę spektralną opartą o dyskretną transformację Fouriera, której poddawany jest wygenerowany ciąg. Wyliczane są także współczynniki widma $X(k)$ ze wzoru:

$$X(k) = \frac{1}{N} \sum_{n=1}^{N-1} x(n) e^{j\frac{2\pi kn}{N}} \tag{3}$$

gdzie: $x(n)$ – ciąg wejściowy; $X(k)$ – współczynniki widma sygnału; N – liczba próbek (długość ciągu), a zarazem liczba współczynników widma.

Z uzyskanego ciągu współczynników wybiera się podzbiór, na podstawie którego generuje się ciąg wynikowy, korzystając z odwrotnej dyskretnej transformaty Fouriera. Poszczególne wartości ciągu $x(n)$ wylicza się ze wzoru:

$$x(n) = \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi kn}{N}} \tag{4}$$

Powyższa metoda została zbadana zarówno dla układów kombinacyjnych, jak i sekwencyjnych. Uzyskany przez autorów błąd w porównaniu z ciągiem niepoddanym kompakcji nie przekraczał kilku procent.

Metodę generowania ciągu wejściowego opartą o łańcuchy Markowa opisano w [9]. Ciąg generowany jest na podstawie takich parametrów statystycznych, jak średnie prawdopodobieństwo wejściowe, średnia gęstość przejść oraz przestrzenną korelację wejść. Zaprezentowany algorytm pozwala na bardzo szybkie wygenerowanie żądanego ciągu wejściowego.

Niejednokrotnie wygenerowany lub rzeczywisty ciąg jest bardzo długi, przez co proces symulacji staje się czasochłonny. Istnieją metody pozwalające zmniejszyć długość ciągu nie zmieniając jego własności, a przez to skracające proces symulacji.

W [7] opisano technikę kompakcji opartą o grupowanie i kolejne próbkowanie. Grupowanie polega na wydzieleniu z ciągu grup par wektorów wejściowych posiadających zbliżone wartości odległości, np. z użyciem charakterystyki ładowania–rozładowania pojemności CDC (ang. *charging discharging capacitance*). Następnie odbywa się próbkowanie, które ma na celu wyeliminowanie powtarzających się fragmentów ciągu.

Technika opisana w [6] bazuje na wartości czułości na moc wejść podstawowych. Pobór mocy zależy nie tylko od aktywności przełączania (częstotliwości zmian sygnałów wejściowych z jedynki na zero lub odwrotnie), ale również od zmienności aktywności przełączania wejść układu. W [1] wprowadzono pojęcie czułości na moc wejść podstawowych $S_{a(x)}$, będącej miarą wpływu zmienności aktywności przełączania wejść na moc, opisanej jako:

$$S_{a(x_i)} = \lim_{\Delta a(x_i)\to 0} \frac{\Delta Power(x_i)}{\Delta a(x_i)} = \frac{\partial Power(x_i)}{\partial a(x_i)} \tag{5}$$

gdzie: $a(x_i)$ jest aktywnością przełączania wejścia xi, $\Delta a(x_i)$ to zmienność aktywności przełączania, a $\Delta Power(x_i) = Power[a(x_i) - \Delta a(x_i)] - Power[a(x_i)]$ jest zmiennością pobieranej mocy dla zadanej zmienności aktywności przełączania $\Delta a(x_i)$.

Techniki kompakcji oparte o fraktale przedstawiono w pracach [13] oraz [14]. Fraktalami nazywa się kształty zbudowane z fragmentów podobnych do jego całości. Ciągi wejściowe również mogą składać się z podobnych do siebie fragmentów, w związku z czym można je porównywać do fraktali.

Wszystkie wymienione metody wymagają przeprowadzania obliczeń arytmetycznych, przez co czas niezbędny do ich realizacji może być dość długi. Alternatywą, która jeszcze nigdy nie była brana pod uwagę przy kompakcji ciągu wejściowego, jest zastosowanie zbiorów przybliżonych, których metody w znacznej mierze opierają się na operacjach na funkcjach boolowskich. Pozwala to na zastosowanie np. układów FPGA do przeprowadzenia kompakcji, co znacząco przyspiesza otrzymywanie wyniku.

4. Zbiory przybliżone i redukt

Zbiory przybliżone stanowią uzupełnienie rachunku prawdopodobieństwa i zbiorów rozmytych i służą wraz z nimi do pozyskiwania wiedzy z niepełnych i nieprecyzyjnych zbiorów danych. Teoria ta powstała w latach 80. XX wieku, a jej twórcą był prof. Zdzisław Pawlak [15].

Pojęcie zbioru przybliżonego rozszerza pojęcie klasycznego zbioru o niepewność, przez co doskonale nadaje się m.in. do klasyfikacji w przypadku nieprecyzyjnych i niepełnych danych.

Podstawowymi pojęciami określającymi granice zbioru przybliżonego są dolna ($\underline{B}X$) i górna aproksymacja ($\overline{B}X$)[16]. Wszystkie obiekty x_i, które na pewno należą do zbioru przybliżonego X, stanowią dolną aproksymację. Dolną aproksymację systemu informacyjnego $SI = \{U, A\}$, gdzie U – zbiór obiektów $\{x_1, \ldots, x_N\}$, natomiast A – zbiór atrybutów $\{a_1, \ldots, a_M, d\}$ wraz z atrybutem decyzyjnym d, można wyznaczyć, korzystając z klas abstrakcji $IND_{SI,B}(x)$. Klasy abstrakcji określane są za pomocą poniższego równania:

$$IND_{SI,B}(x) = \{y \in U : (x, y) \in IND_{SI}(B)\} \tag{6}$$

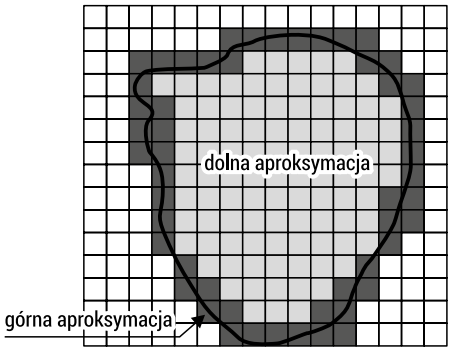
gdzie: $IND_{SI,B}(x)$ – klasa nierozróżnialności względem atrybutów ze zbioru B; $IND_{SI}(B) = \{(x, y) : \forall_{a \in B} d(x) = d(y)\}$ – relacja nierozróżnialności względem atrybutów ze zbioru B [16].

W takim przypadku dolna aproksymacja jest określona wzorem:

$$\underline{B}X = \{x \in U : IND_{SI,B}(x) \subseteq X\} \tag{7}$$

Wszystkie obiekty x_i, które prawdopodobnie należą do zbioru przybliżonego X, stanowią górną aproksymację [15]. Można ją również wyliczyć, korzystając z klas nierozróżnialności za pomocą zależności:

$$\overline{B}X = \{x \in U : IND_{SI,B}(x) \cap X \neq \varnothing\} \tag{8}$$



RYS. 3. Górna i dolna aproksymacja zbioru przybliżonego
FIG. 4. Lower and upper approximation of a rough set

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Na rysunku 3 przedstawiono schematycznie dolną i górną aproksymację zbioru przybliżonego. Dolną aproksymację stanowi jasnoszary obszar wewnątrz obramowania oznaczającego zbiór przybliżony. Górną aproksymacją są jasno- i ciemnoszare elementy, przy czym ciemnoszare elementy są nazywane obszarem granicznym.

Nie wszystkie atrybuty ze zbioru A są niezbędne do zachowania pełnej rozróżnialności obiektów ze zbioru X. Dla danego systemu informacyjnego IS może istnieć podzbiór $R \subseteq A$, który pozwala na taki sam podział obiektów na klasy decyzyjne jak pełny zbiór atrybutów A. Podzbiór R nazywany jest reduktem [5].

Dla danego IS może istnieć więcej niż jeden redukt, jak również może nie być żadnego reduktu. Jednocześnie przy więcej niż jednym redukcie może istnieć pewna grupa atrybutów C, która jest fundamentalna i występuje w każdym z nich. Grupa atrybutów C nazywana jest rdzeniem. Żaden redukt nie może nie zawierać wszystkich atrybutów znajdujących się w rdzeniu [5].

Czasami atrybut może przyjmować wartości ciągłe lub dyskretne, ale z bardzo licznego zbioru. Może to powodować pewne utrudnienia przy klasyfikacji oraz niepotrzebnie zwiększać rozmiar tablicy decyzyjnej. W związku z tym często stosuje się dyskretyzację, czyli proces zmniejszania liczności zbioru wartości danego atrybutu [5]. Należy jednak dochować wszelkich starań, aby proces ten miał jak najmniejszy wpływ na podejmowane decyzje.

5. Proponowany algorytm oraz badania

Istnienie dwóch przekształceń tablicy decyzyjnej, tzn. redukcji (wyznaczanie reduktu i usuwanie atrybutów nieznajdujących się w redukcie) oraz dyskretyzacji (zmniejszanie liczności zbioru wartości danego atrybutu w taki sposób, żeby w jak najmniejszym stopniu wpłynąć na decyzję końcową) pozwala wysnuć wniosek, iż można je zastosować do kompakcji wejściowego strumienia bitowego. Do tego celu należy przygotować tablicę decyzyjną składającą się z atrybutów warunkowych, atrybutu decyzyjnego oraz obiektów (tab. 1).

TAB. 1. Tablica decyzyjna
TAB. 1. Decision table

		Atrybuty						
		a_1	a_2	a_3	a_4	...	a_N	d
Obiekty	x_1							
	x_2							
	...							
	x_M							

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Tablica decyzyjna z tabeli 1 składa się *N* atrybutów warunkowych, jednego decyzyjnego oraz *M* obiektów. W przypadku strumienia bitowego można przyjąć, że atrybutem jest każdy z bitów poszczególnych słów wejściowych. Dodatkowo obiektami będą słowa strumienia bitowego. Do przeprowadzenia redukcji oraz dysktretyzacji trzeba jeszcze wprowadzić do tablicy decyzyjnej atrybut decyzyjny. W normalnych warunkach atrybut decyzyjny określa decyzję podjętą na podstawie wartości atrybutów warunkowych. W przypadku ciągu bitowego decyzja będzie polegała

na określeniu, do którego stanu nastąpi przejście w momencie pojawienia się na wejściu układu słowa ze strumienia wejściowego. Do realizacji tego zadania konieczna jest znajomość architektury układu sekwencyjnego, dla którego przeprowadzana jest symulacja. Jednym z najczęściej spotykanych opisów układu sekwencyjnego jest lista przejść. W liście przejść (tab. 2) każdy element listy składa się z czterech wartości:

- stanu bieżącego a_i – stanu układu przed przejściem;
- słowa wejściowego $x(a_i \rightarrow a_i')$, które powoduje przejście ze stanu a_i do stanu następnego;
- słowa wyjściowego $y(a_i')$, które pojawi się na wyjściu przy przejściu do stanu następnego;
- stanu następnego a_i', do którego przejdzie układ po pojawieniu się na wejściu słowa $x(a_i \rightarrow a_i')$.

TAB. 2. Lista przejść układu sekwencyjnego
TAB. 2. Transition list of a sequential circuit

Stan bieżący	Słowo wejściowe	Słowo wyjściowe	Stan następny
a_1	$x(a_1 \rightarrow a_1')$	$y(a_1')$	a_1'
a_2	$x(a_1 \rightarrow a_1')$	$y(a_1')$	a_1'
...	...	...	...
al.	$x(a_L \rightarrow a_L')$	$y(a_L')$	a_L'

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Biorąc pod uwagę kolumnę drugą i czwartą z listy przejść z tabeli 2, można łatwo uzupełnić tablicę decyzyjną o atrybut decyzyjny. Gdy tablica decyzyjna zostanie przygotowana można przystąpić do jej redukcji. Do generowania reduktu można zastosować algorytm opisany w [5], gdzie bazuje się na częstości wystąpień danego atrubutu w tzw. macierzy rozróżnialności. Natomiast dyskretyzację można zrealizować za pomocą algorytmu opisanego w [8], gdzie stosuje się tzw. dyskretyzację z maksymalną rozróżnialnością. Dyskretyzację należy wykonać, zakładając, że atrybutem warunkowym jest całe słowo, a nie jak w przypadku generowania reduktu poszczególne bity. Jest to spowodowane tym, że nie da się dokonać dyskretyzacji atrybutu, który może przyjmować tylko dwie wartości.

Poniżej przedstawiono opis proponowanego algorytmu kompakcji strumienia wejściowego.

Algorytm kompakcji strumienia wejściowego

dane: wejściowy strumień bitowy $S = \{s_i\}$, lista przejść $T = \{< a_i, x(a_i \rightarrow a_i'), y(a_i'), a_i' >\}$
wyjście: skompaktowany strumień S'

1. Przygotuj tablicę decyzyjną D składającą się z N+1 kolumn, gdzie N jest liczbą wejść układu sekwencyjnego, oraz M wierszy, gdzie M jest liczbą słów w strumieniu wejściowym S.
2. Dla każdego wiersza k tablicy decyzyjnej D:
   a) w kolumnie N+1 umieść wartość a_i' z listy przejść T, dla której słowo s_k strumienia S ma wartość identyczną z $x(a_i \rightarrow a_i')$.
3. Wykonaj procedurę generowania reduktu R dla tablicy decyzyjnej D zgodnie z poniższą procedurą (na bazie [5]):
   a) stwórz kopię tablicy decyzyjnej D, na której wykonasz poniższe operacje;
   b) dla każdej pary obiektów należących do różnych klas decyzyjnych dla każdego atrybutu a_i oblicz liczbę wystąpień różnicy wartości atrybutu między elementami pary;
   c) wybierz atrybut a_{max} z największą liczbą wystąpień i dodaj go do reduktu;
   d) usuń wszystkie obiekty, które należały do par wskazujących na konieczność wybrania atrybutu a_{max};
   e) jeżeli w tablicy pozostają jakieś obiekty, to wróć do punktu b.
4. Scal wszystkie bity (atrybuty warunkowe) w tablicy decyzyjnej w jeden atrybut warunkowy.
5. Wykonaj procedurę dyskretyzacji zgodnie z poniższym opisem (na bazie umieszczonego w [8]):
   a) posortuj wartości atrybutów w porządku rosnącym;
   b) podziel atrybuty na dwie części w taki sposób, aby liczba obiektów należących do różnych klas decyzyjnych w każdej z części była maskymalna;
   c) dla każdej z części wykonaj ponowny podział zgodnie z punktem b);
   d) przerwanie pęti następuje w momencie, gdy liczba obiektów należących do różnych klas decyzyjnych w każdej z części osiągnie założoną wartość minimalną.
6. W każdej z części pozostaw po jednym (pierwszym) obiekcie.

Badania algorytmu zostały przeprowadzone dla losowego strumienia wygenerowanego przez funkcję *rand*() z biblioteki *glibc* (środowisko MinGW dołączone do IDE Code::Blocks w wersji 17.12). Strumień składał się z 1024 słów o rozmiarze 8 bitów. Strumień wejściowy został poddany kompakcji dla układów sekwencyjnych ze zbioru [18]. Wyniki badań przedstawiono w tabeli 1. W kolumnie „Benchmark" umieszczono nazwę układu, kolumna „Długość" zawiera długość strumienia po procesie kompakcji, natomiast kolumna „% redukcji" opisuje o ile procent zmniejszył się strumień po procesie kompakcji.

TAB. 3. Lista przejść układu sekwencyjnego
TAB. 3. Transition list of a sequential circuit

Benchmark	Długość	% redukcji
bbara	936	8,6
bbsse	818	20,1
bbtas	832	18,8
beecount	835	18,5
keyb	880	14,1
s27	910	11,1
tav	881	14,0

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Wyniki przedstawione w tabeli 3 potwierdzają, iż proponowany algorytm można wykorzystać do kompakcji ciągu. Największą redukcję uzyskano dla układu „bbsse", natomiast najgorsze wyniki – dla układu „bbara". Średnia wartość redukcji wyniosła 15%.

6. Podsumowanie i wnioski

Przedstawione rozwiązanie stanowi przykład nowego podejścia do procesu kompakcji oraz wskazuje możliwość zastosowania metod zbiorów przybliżonych do zmniejszania długości wejściowych strumieni bitowych.

Badania eksperymentalne potwierdziły, że możliwa jest redukcja rozmiaru strumienia bitowego z wykorzystaniem algorytmu bazującego na teorii zbiorów przybliżonych, a stopień redukcji, wynoszący 15%, pozwala mieć nadzieję, iż po dokonaniu usprawnień algorym może mieć praktyczne zastosowanie w symulacji poboru mocy układów sekwencyjnych.

Przyszłe prace powinny skupić się na sprawdzeniu możliwości innego podziału bitów ciągu na atrybuty (w tym wielobitowe), innego sposobu przypisania atrybutów decyzyjnych bądź zastosowania deyskretyzacji wielowymiarowej (dla wielu atrybutów).

Literatura

1. Chen Z, Roy K, Chou TL. *Power Sensitivity – a New Method to Estimate Power Dissipation Considering Uncertain Specifications of Primary Inputs*. Proceedings of the 1997 IEEE/ACM International Conference on Computer Aided Design, San Jose, CA, 1997; 40-44.
2. Ghosh A, Devedas S, Kreutzer K, White J. *Estimation of Average Switching Activity in Combinational and Sequential Circuits*. Proc. of 29th ACM/IEEE Design and Automation Conference, 8-12 June 1992; 253-259.
3. Grześ T, Salauyou V. Metody obliczania mocy w układach cyfrowych. *Pomiary, Automatyka, Kontrola*. 2006; 7bis: 101-102.
4. Grześ T. *Sequential Circuits Power Modeling for Low Power Design*. Proceedings of XVI Ukrainian-Polish Conference „CAD in Machinery Design. Implementation and Educational Problems" (CADMD'2006), Polyana, Ukraine, May 22-23, 2006; 54-56.
5. Grześ T, Kopczyński M, Stepaniuk J. FPGA in rough set based core and reduct computation. *Lecture Notes in Artificial Intelligence*. 8171: 263-270.
6. Hsu C-Y, Wei C-W, Shen W-Z. *A Pattern Compaction Technique for Power Estimation Based On Power Sensitivity Information*. Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), Sydney, Australia, May 2001, vol. 5; 467-470.
7. Hsu C-Y, Shen W-Z. *Vector Compaction for Power Estimation with Grouping and Consecutive Sampling Techniques*. IEEE International Symposium on Circuits and Systems, Scottsdale, Arizona (USA), May 2002; II-472 – II-475.
8. Kopczyński M, Grześ T, Stepaniuk J. Maximal Discernibility Discretization of Attributes – a FPGA Approach. *Studies in Big Data*. 2016; 19: 171180.
9. Liu X, Papaefthymiou MC. A Markov Chain Sequence Generator for Power Macromodeling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 2004; vol. 23, no. 7: 1048-1062.
10. Macii A, Macii E, Poncino M, Scarsi R. Stream Synthesis for Efficient Power Simulation Based on Spectral Transforms. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*. 2001; vol. 9, no. 3: 417-426.
11. Pedram M. Design technologies for Low Power VLSI. *Encyclopedia of Computer Science and Technology*. 1995; 36.
12. Pedram M. Power simulation and estimation in VLSI circuits. In: Chen W-K, ed. *The VLSI Handbook*. The CRC Press and the IEEE Press; 1999.
13. Radjassamy R, Carothers JD. *A Fractal Compaction Algorithm for Efficient Power Estimation*. Proceedings of the International Conference on Computer Design: VLSI in Computers and Processors ICCD '98, 5-7 Oct 1998, Austin, TX, USA; 542-547.

14. Radjassamy R, Carothers D. Faster Power Estimation of CMOS Designs Using Vector Compaction – A Fractal Approach. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*. 2003; vol. 33, no. 3: 476-488.
15. Stepaniuk J. Knowledge discovery by application of rough set models. In: Polkowski L, Tsumoto S, Lin TY, eds *Rough Set Methods and Applications. New Developments in Knowledge Discovery in Information Systems*. Heidelberg: Physica-Verlag; 2000;137-233.
16. Stepaniuk J. *Rough-Granular Computing in Knowledge Discovery and Data Mining*. Springer; 2008.
17. Tsui C-Y, Monteiro J, Pedram M, Devadas S, Despain AM, Lin B. Power Estimation Methods for Sequential Logic Circuits. *IEEE Transactions on VLSI Systems*. 1995; vol. 3, no. 3: 404-416.
18. Yang S. Logic Synthesis and Optimization Benchmarks User Guide: Version 3.0. „Technical Report", Microelectronics Center of North Carolina, 1991; 43.

Badania zostały zrealizowane w ramach pracy nr S/WI/1/2018 i sfinansowane ze środków na naukę Ministerstwa Nauki i Szkolnictwa Wyższego.

Streszczenie

Minimalizacja mocy i projektowanie układów o obniżonym poborze mocy to bardzo ważne trendy w projektowaniu współczesnych układów cyfrowych. Projektowanie takie wymaga obliczenia mocy pobieranej przez system, co można uzyskać za pomocą symulacji. Proces ten może być bardzo długi, ponieważ wymaga wejściowego strumienia bitowego.

Kompresja bitowego strumienia danych jest jedną z technik wykorzystywanych do usprawniania symulacji poboru mocy układów sekwencyjnych. Do tego celu nigdy nie były stosowane metody zbiorów przybliżonych. Niniejsze opracowanie jest pierwszym podejściem do kompakcji danych przy użyciu zbiorów przybliżonych.

Słowa kluczowe: układy sekwencyjne, symulacja mocy, kompakcja, zbiory przybliżone

Summary
Bitstream compaction using rough set theory for power simulation in sequential circuits

Power minimization and low power design are very important trends in digital circuits design. Minimizing power consumption, for example, leads to increasing the time be-tween recharging the battery in mobile systems. It is also needed to design environmen-tally friendly devices. Low-power designing and power-aware designing, in one of the stages, need to calculate the power consumed by the system. This can be obtained by power simulation, but this process can be very long, as it needs the stream of data that is given to inputs of the simulated circuit. The stream may be compacted to cut the simulation time. The rough sets theory was developed by Prof. Z. Pawlak

in eighties of the 20th century. Rough sets are used for data analysis and processing, and one of the fields of its utilization is a data reduction. Removing the redundancy of the data stream in the rough sets can be achieved with the reduct and the discretization.

In this study, the reduct is used for data stream compaction. Proposed algorithm need two informations: the data stream and the transition list of the sequential circuit. Data stream is treated as the decision table, extended by the data extracted from the transi-tion list to form the decision attribute.

In proposed solution data stream is treated as the decision table – any bit of the data stream represent one conditional attribute. To complete the decision table there is deci-sion attribute needed. Decision attribute is generated from the transition list of the sequ-ential circuit. Full decision table is then processed to obtain the reduct and to perform the discretization. These two processes provide data reduction and the resulting data stream is significantly smaller, that lead to decrease the power simulation time.

Data stream compaction is one of the techniques used in improving the power simula-tion of the sequential circuits, but never used rough sets methods. This work is first approach to data compaction/compression using rough sets methods.

Keywords: sequential circuits, power simulation, compaction, rough sets

DOI: 10.24427/978-83-66391-28-4_8

Problem komunikacji federatów w symulacji opartej o HLA

Krzysztof PANUFNIK[*]

1. Wprowadzenie

Standard HLA (ang. *High Level Architecture*) powstał z myślą o łączeniu ze sobą różnorodnych symulatorów [1], zwanych w standardzie federatami, bez względu na technologię, w której zostały wykonane. Dzięki połączeniu federatów za pomocą HLA możliwe jest tworzenie rozbudowanych symulacji (zwanych federacjami), w których symulatory współdzielą informacje o obiektach symulacyjnych. HLA jest standardem, który realizowany jest za pomocą RTI (ang. *Run Time Infrastructure*).

Wraz ze wzrostem liczby federatów podłączonych do HLA oraz symulowanych obiektów wzrasta problem dystrybucji informacji o zmianach w tych obiektach do federatów. HLA definiuje dwie usługi, których można użyć w zależności od skali symulacji. Mają one za zadanie ograniczenie liczby wysyłanych zmian.

Dużej liczbie zmian obiektów towarzyszy problem sposobu wykorzystania zasobów sieciowych do dystrybucji zmian. Implementacje HLA do dystrybucji danych w sieci lokalnej wykorzystują grupy multicast protokołu IP. Liczba takich grup może okazać się niewystarczająca w przypadku dużych eksperymentów symulacyjnych. Potrzebny jest wtedy algorytm podziału federatów na grupy, który zapewni wydajną komunikację.

2. Usługi filtrowania danych HLA

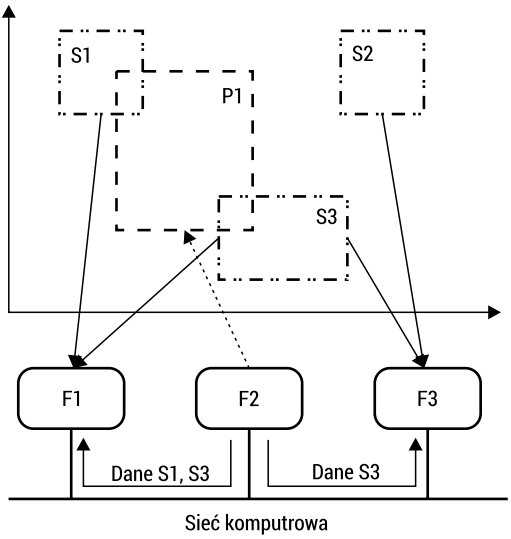
HLA przewiduje istnienie dwóch usług, których celem jest ograniczenie liczby przesyłanych informacji. Są to Zarządzanie Deklaracjami (ang. *Declaration Management*) oraz Zarządzanie Dystrybucją Danych (ang. *Data Distribution Management*) [2]. Obie te usługi działają, wykorzystując wzorzec publikuj–subskrybuj. Federaty mogą zadeklarować, jakie obiekty zamierzają zmieniać lub o zmianach w jakich obiektach chcą być informowane.

[*] Wojskowa Akademia Techniczna

Usługa DM pozwala na definiowanie zainteresowania poprzez wskazanie typu obiektu, np. pojazdu, czołgu czy motocykla. W połączeniu z możliwościami modelowania obiektów symulacyjnych (wprowadzania relacji „jest" między obiektami) możliwe jest zadeklarowanie zainteresowania jedną grupą obiektów, co przełoży się na zainteresowanie grupami obiektów, które są związane relacją „jest" ze wskazaną grupą.

Usługa DDM daje dużo większe możliwości wyrażania zainteresowań [3]. Jest to możliwe dzięki wprowadzeniu filtrowania na poziomie atrybutów obiektów. Podczas projektowania modelu obiektów symulacyjnych definiowane są atrybuty, po których może odbywać się filtrowanie, oraz wartości, jakie atrybuty te mogą przyjmować. Atrybut taki nazywany jest wymiarem, a zbiór jego wartości tłumaczony jest na dodatnie wartości liczbowe. Po zdefiniowaniu wszystkich atrybutów filtrujących możliwe jest stworzenie wielowymiarowej przestrzeni (o liczbie wymiarów równej liczbie atrybutów filtrujących). W tak zdefiniowanej przestrzeni wprowadza się następujące pojęcia:

- zakres – przedział będący podzbiorem wartości wymiaru,
- obszar – wielowymiarowa bryła powstała przez zdefiniowanie niepustych zakresów dla każdego z wymiarów.



RYS. 1. Działanie usługi DDM
FIG. 1. DDM service

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Działanie usługi DDM przebiega w czterech fazach:

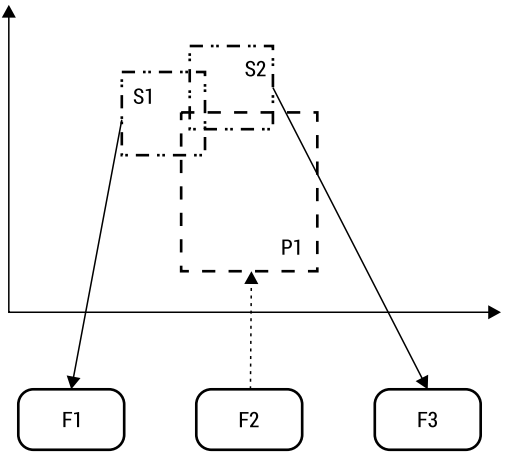
- deklarowanie – federaci deklarują zainteresowanie danymi poprzez wskazanie jednego lub więcej obszarów wraz ze wskazaniem, jakiego typu jest do obszar (związany z subskrypcją lub publikacją);

- wyszukiwanie przecięć – poszukiwanie części wspólnych między obszarami publikującymi a obszarami subskrybującymi,
- połączenie federatów – wskazanie kanałów komunikacji, którymi federaty mają się posługiwać w celu wymiany danych związanych z poszczególnymi obszarami danych;
- wymiana danych – przesłanie danych, zgodne z wcześniej ustalonymi kanałami komunikacji.

Na rysunku 1 przedstawiono przykład działania usługi DDM dla federacji, w skład której wchodzą trzy federaty. Zdefiniowano dwa atrybuty filtrujące, co spowodowało stworzenie dwuwymiarowej przestrzeni. W przestrzeni tej zostały zadeklarowane następujące regiony:

- subskrypcji (S1, S2 oraz S3) – federaty F1 i F3 są zainteresowane zmianami obiektów znajdujących się odpowiednio w obszarach S1 i S3 dla federata F1 oraz S2 i S3 dla federata F3;
- publikacji (P1) – federat F2 zadeklarował zmienianie obiektów znajdujących się w tym obszarze.

Region P1 przecina się z regionami S1 i S3. W rezultacie federat F2 będzie wysyłał dane o zmianach w regionie P1 (a więc i części regionów S1 i S3) do federatów F1 i F3. W ten sposób federaty F1 i F3 otrzymają wszystkie zmiany, którymi były zainteresowane. Federat F3 nie otrzyma informacji o zmianach w obszarze S2, gdyż nie ma federata, który by zmieniał dany obszar.



RYS. 2. Przecięcie obszarów zainteresowań
FIG. 2. Regions intersection

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Należy wspomnieć o pewnych ograniczeniach, które zostały poczynione przy tworzeniu wymagań dla usługi DDM. Przyjęto, że zakresy są przedziałem dopuszczalnych wartości wymiaru. To w połączeniu ze sposobem definiowania obszarów, tj. przez podanie zakresów, które je definiują, skutkuje tym, że obszary mogą przyjmować jedynie kształt hiperprostokątów. Często zdarza się, że przecięcia obszarów nie są hiperprostokątami. Taką sytuację przedstawia rysunek 2.

Można wyróżnić istnienie trzech przecięć, których zmiany powinny być rozsyłane do federatów:

$$I1 = S1 \cap P1 \setminus S2$$

$$I2 = S2 \cap P1 \setminus S1$$

$$I3 = S1 \cap S2 \cap P1$$

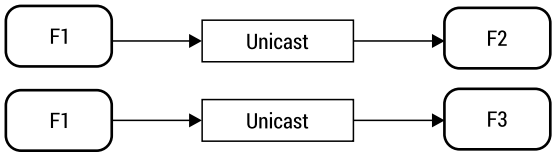
Do federata F1 powinny zostać wysłane zmiany z przecięć I1 oraz I3, a do federata F2 – I2 oraz I3. Ze względu na ograniczenia przyjęte w standardzie HLA tak się nie stanie, a oba federaty otrzymają informacje o wszystkich zmianach w obszarze P1, nawet tych, którymi nie byli zainteresowani. Ograniczenie to zostało najprawdopodobniej wprowadzone, aby ułatwić poszukiwanie przecięć między obszarami.

Powyższy problem można częściowo rozwiązać poprzez zdefiniowanie bardzo wielu małych regionów publikacji i subskrypcji. Miałoby to jednak negatywny wpływ na proces odnajdywania przecięć między obszarami. W dalszej części rozważać się będzie obszary i przecięcia o dowolnych kształtach.

Deklaracje obszarów mogą zmieniać się w czasie. W przypadku zmiany wymagane jest ponowne ustalenie kanałów do komunikacji federatów.

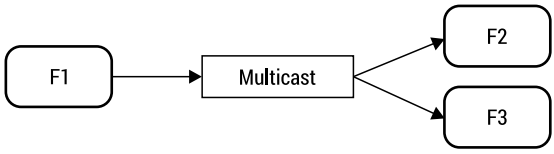
3. Kanały komunikacji w sieci komputerowej

Najczęstszą metodą poprawy wydajności jest wykorzystanie sieciowych grup multicast. Jest to jedna z trzech metod komunikacji w sieci komputerowej. Pozostałe dwie to unicast, czyli połączenie „jeden do jeden", oraz broadcast, czyli połączenie „wszyscy do wszystkich". Połączenia typu multicast pozwalają na wysyłanie danych tylko do komputerów będących członkami grupy multicast. W efekcie komputer wysyłający dane może wysłać je jednokrotnie. Alternatywą dla tego rozwiązania byłoby wysyłanie do każdego z odbiorców danych za pomocą połączenia unicast, co skutkowałoby wielokrotnym przesyłaniem tych samych informacji. Dzięki połączeniom multicast możliwe jest więc realizowanie komunikacji typu „jeden do wielu" oraz „wiele do wielu". Na rysunkach 3, 4 oraz 5 przedstawiono możliwe sposoby wykorzystania kanałów multicast i unicast do komunikacji.



RYS. 3. Komunikacja z wykorzystaniem kanałów unicast
FIG. 3. Unicast communication

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 4. Komunikacja typu „jeden do wielu" z wykorzystaniem kanału multicast
FIG. 4. One to many multicast communication

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

RYS. 5. Komunikacja typu „wiele do wielu" z wykorzystaniem kanału multicast
FIG. 5. Many to many multicast communication

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Choć komunikacja za pomocą kanałów multicast wydaje się odpowiadać na potrzeby dystrybucji danych w symulatorach opartych o HLA, nie jest ona pozbawiona wad. Główną jest ograniczona liczba grup multicast, które można utworzyć. Choć istnieją urządzenia, które pozwalają na zdefiniowanie 1024-2000 grup multicast [4][5], nierzadko spotyka się wartości 256 lub mniejsze [6, 7]. Wartości te mogą być niewystarczające dla złożonych symulacji o dużej liczbie obszarów.

Tworzenie nowych grup multicast wiąże się ze zbudowaniem drzewa routingu między nadawcą a odbiorcami. W zależności od wybranego algorytmu komunikacji multicast koszt stworzenia drzewa, jak i jego wpływ na ogólną wydajność sieci, może być różny [8]. Zdefiniowanie dużej liczby grup multicast stanowi więc obciążenie dla sieci, nawet gdy grupa nie jest wykorzystywana. Czas potrzebny na dołączenie

lub opuszczenie grupy multicast również może być znaczący (od kilkuset milisekund do kilku sekund [9]). W związku z tym wykorzystanie grup multicast sprawdza się najlepiej w przypadku symulacji, w której nie występuje wiele zmian obszarów.

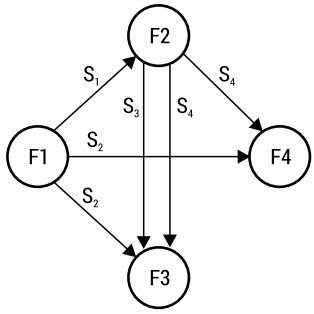
4. Dotychczasowe rozwiązania

Dotychczasowe metody rozwiązania problemu podziału federatów na grupy multicast można podzielić na bazujące na:

- regionach – sprawdzane są wszystkie przecięcia regionów publikujących ze wszystkimi regionami subskrybującymi, a grupy multicast są przypisywane w przypadku wystąpienia przecięcia;
- siatce – przestrzeń dzieli się dodatkowo za pomocą siatki o stałym lub zmiennym rozmiarze komórki; następnie danej komórce przypisuje się grupę multicast, w zależności od przyjętej metody zawsze lub tylko wtedy, gdy w danej komórce występują przecinające się regiony.

Do rozwiązań opartych o regiony należą algorytmy LOC (ang. *Largest Outgoing Connection*), IRLOC (ang. *Input-Restricted* LOC) [10] oraz ACPS (ang. *Adaptive Communication Protocol Selection*).

Dla każdego regionu publikującego, dla którego zachodzi przecięcie z regionem subskrybującym, definiuje się strumień zmian. Federaty subskrybujące, których regiony przecięły się z regionem publikującym, są odbiorcami strumienia zmian. Zależności między federatami definiowane przez strumienie można przedstawić za pomocą grafu strumieni zmian (rys. 6).



RYS. 6. Graf strumieni zmian
FIG. 6. A change stream graph
ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Ze strumieniem zmian skojarzone są więc: federat, waga strumienia będąca sumą rozmiarów wszystkich zmian wysłanych przez federata z danego obszaru oraz zbiór federatów, do których należy wysłać zmiany. Zbiór strumieni zmian dla grafu z rysunku 6 to:

$$S = \left\{ \left\langle f_1, w_1, \left\{ f_2 \right\} \right\rangle, \left\langle f_1, w_2, \left\{ f_3, f_4 \right\} \right\rangle, \left\langle f_2, w_3, \left\{ f_3 \right\} \right\rangle, \left\langle f_2, w_4, \left\{ f_3, f_4 \right\} \right\rangle \right\}$$

Mając dany zbiór strumieni zmian S, algorytmy dzielą strumienie zmian między m grup multicastowych tak, aby czas przesłania wszystkich zmian był jak najmniejszy, nie większy niż t_{max}. Mając następujące dane:

- zbiór federatów:

$$F = \left\{ f_1, f_2, \ldots f_n \right\}, f_i \in N$$

- macierz przepustowości łącza między federatami:

$$C = \left[c_{i,j} \right] c_{i,j} \in N, i, j \in F$$

- funkcję opisującą liczbę zmian wysłaną przez federata:

$$gs(x) \in N, x \in F$$

- czas wysyłki pojedynczej zmiany przez federata:

$$ts_i \in N, i \in F$$

- funkcję opisującą liczbę zmian odebraną przez federata:

$$gr(x) \in N, x \in F$$

- czas odbioru pojedynczej zmiany przez federata:

$$tr_i \in N, i \in F$$

- funkcję opisującą liczbę bajtów przesłanych od federata x_1 do federata x_2:

$$gt(x_1, x_2) \in N, x_1, x_2 \in F$$

- czas przesłania wszystkich wiadomości daje się oszacować jako:

$$\max \left\{ gs(i) \cdot ts_i + \frac{gt(i,j)}{c_{i,j}} + gr(j) \cdot tr_j : i, j \in F \right\}$$

Algorytm LOC sortuje listę strumieni na podstawie ich wag. Poczynając od strumienia o największej wadze algorytm sprawdza, czy możliwe jest dodanie nadawcy i odbiorców strumienia do grupy multicast. Jeżeli dodanie strumienia do grupy spowodowałoby przekroczenie t_{max}, sprawdzana jest możliwość dodania strumienia do kolejnej grupy. W przypadku, gdy nie ma więcej grup, strumień jest rozsyłany do wszystkich odbiorców za pomocą połączeń unicast. Wadą algorytmu jest to, że skupia się on wyłącznie na minimalizacji obciążenia federatów wysyłających.

Powyższy problem rozwiązuje algorytm IRLOC. Wprowadza on dodatkową miarę strumienia, jaką jest zysk z użycia połączenia multicast:

$$\left(\mathrm{k}_i - 1\right) \cdot w_i$$

gdzie k_i jest liczbą federatów odbierających i-tego strumienia. Miara opisuje, ile bajtów mniej zostanie wysłanych w przypadku użycia połączenia multicast zamiast rozsyłania wiadomości wielokrotnie połączeniem unicast. Dołączenie strumienia do grupy multicast może przynieść negatywny skutek w postaci zwiększenia się liczby nadmiarowych wiadomości u federatów:

- będących już członkami grupy multicast, ze względu na to, że nie są odbiorcami dodawanego strumienia;
- będących odbiorcami dodawanego strumienia, ale nie będących odbiorcami jednego bądź większej liczby strumieni już dodanych do grupy.

Algorytm IRLOC wprowadza dwie zmiany w stosunku do algorytmu LOC:

- strumienie są sortowane nie ze względu na wagę, ale ze względu na zysk z użycia połączenia multicast;
- strumień jest dodawany do grupy tylko wtedy, gdy jego zysk z użycia połączenia multicast przewyższa negatywny skutek.

Wadą algorytmu IRLOC jest to, że nie uwzględnia on faktu, że strumienie mogą mieć podobną grupę federatów odbierających.

Problem ten rozwiązuje algorytm ACPS. Wprowadza on miarę odległości między strumieniem a grupą multicast. Miarą tą jest negatywny skutek dodania strumienia do grupy. W stosunku do algorytmu IRLOC zmienia się wybór grupy, do której należy dodać strumień. W algorytmie IRLOC grupy były sprawdzane po kolei, natomiast w algorytmie ACPS wybierana jest ta grupa, dla której odległość strumienia jest najmniejsza.

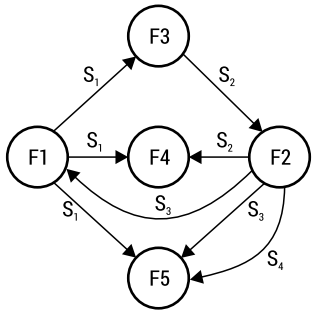
5. Koncepcja rozwiązania problemu

Opisane algorytmy podejmowały decyzję o dodaniu strumienia do grupy multicast na podstawie jednego bądź obu kryteriów minimalizacji obciążenia federatów wysyłających i odbierających. Oba te kryteria wpływają na czas przesłania wszystkich wiadomości. Omówione algorytmy nie biorą jednak pod uwagę, że federaty nie są sobie równe. Dzieje się tak dlatego, że podstawą podejmowania decyzji są strumienie zmian. Czas transmisji jest ograniczony od góry przez parę federatów najbardziej obciążonych. Traktowanie podziału z punktu widzenia strumieni zmian nie daje możliwości odciążenia federatów z par najbardziej obciążonych. W celu rozwiązania tego problemu proponuje się wprowadzenie następujących zmian:

- podział strumieni na mniejsze poprzez rozdzielenie federatów odbierających;

- inny sposób sortowania strumieni, który zapewni najbardziej obciążonym federatom pierwszeństwo w przydziale do grup multicast;
- wybór grupy, do której należy dodać strumienie powinien być motywowany minimalizacją obciążenia najbardziej obciążonych par federatów, co doprowadziłoby do zmniejszenia całkowitego czasu przesłania.

Przykład eksperymentu symulacyjnego, dla którego zaproponowane zmiany mogą poprawić wydajność komunikacji, przedstawiono na rysunku 7. Parametry strumieni zostały natomiast zaprezentowane w tabeli 1.



RYS. 7. Graf strumieni zmian
FIG. 7. A change stream graph

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Zakłada się następujące parametry:
a) czas wysłania i odebrania pojedynczej zmiany przez federata jest taki sam dla wszystkich federatów i wynosi 1ms,
b) czas przesłania zmiany między dowolną parą federatów wynosi 2ms,
c) dostępna liczba grup multicast równa jest 2.

TAB. 1. Parametry strumieni zmian
TAB. 1. Change streams parameters

Strumień	Nadawca	Odbiorcy	w_i	$(k_i - 1) \cdot w_i$
S_1	1	3,4,5	5	10
S_2	2	3,4	5	5
S_3	2	1,5	4	4
S_4	2	5	3	0

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

TAB. 2. Przypisanie strumieni zmian do kanałów komunikacji
TAB. 2. Assignment of change streams to communication channels

Algorytm	Grupa 1	Grupa 2	Połączenia bezpośrednie
IRLOC	<1, S_1,{3,4,5}> <2, S_2,{3,4}>	<2, S_3,{1,5}>	<2, S_4,{5}>
ACPS	<1, S_1,{3,4,5}>	<2, S_2,{3,4}>	<2, S_3,{1}> <2, S_3,{5}> <2, S_4,{5}>
Proponowany algorytm	<1, S_1,{3,4}> <2, S_2,{3,4}>	<2, S_3,{1,5}>	{1, S_1,{5}> <2, S_4,{5}>

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

TAB. 3. Liczba zmian wysyłanych i odbieranych przez federaty oraz czas przesłania
TAB. 3. Number of changes send and received by federates and total transmission time

Federaty	Wysyłające		Odbierające				Czas
	1	2	1	3	4	5	
Minimum	5	12	4	10	10	12	48
IRLOC	5	12	4	10	10	17	53
ACPS	5	16	4	10	10	12	60
Proponowany algorytm	10	12	4	10	10	12	48

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Przypisanie strumieni zmian do kanałów komunikacyjnych, będące wynikiem działania poszczególnych algorytmów, przedstawiono w tabeli 2. Przypisanie ma postać trójki:

(nadawca, strumień, zbiór odbiorców)

Z analizy grafu strumieni zmian wynika, że federat 2 jest tym, który musi rozesłać największą liczbę zmian. Z federatów odbierających federat 5 jest natomiast tym, który musi odebrać największą liczbę zmian. W tabeli 3 przedstawiono wyniki dla poszczególnych przypisań. Algorytm IRLOC spowodował dodatkowe obciążenie federata 5. Algorytm ACPS minimalizuje nadmiary po stronie federatów odbierających, czego efektem było dodatkowe obciążenie dla federata 2. Zaproponowany algorytm nie powoduje dodatkowego obciążenia dla najwolniejszej pary federatów 2 i 5. W efekcie

działania algorytmu dodatkowo obciążony zostaje federat 1. Liczba zmian, które federat miał rozesłać nie jest duża, więc dodatkowe obciążenie nie powoduje pogorszenia całkowitego czasu przesłania zmian.

6. Podsumowanie

W opracowaniu przedstawiono podstawy problemu komunikacji federatów w środowisku HLA oraz główne ograniczenia sieciowe, które wpływają na dobór sposobu komunikacji. Jednocześnie zwrócono uwagę na istotność doboru strategii komunikacji na wydajność eksperymentu symulacyjnego. Zauważono ograniczenia przedstawionych algorytmów i przedstawiono koncepcje zmiany algorytmów celem odciążenia najbardziej obciążonych federatów. Pokazano istnienie eksperymentów symulacyjnych, potwierdzających, że zmiana ta powinna pozytywnie wpłynąć na czas trwania eksperymentu.

Literatura

1. "Introduction to IEEE Std 1516-2010, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules", IEEE 2010.
2. "IEEE Std 1516-2010, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) – Framework and Rules", IEEE 2010.
3. "IEEE Std 1516-2010, IEEE Standard for Modeling and Simulation (M&S) High Level architecture (HLA) Federate interface specification", IEEE2010.
4. https://extremenetworks-ua.com/assets/files/SummitX650/DSSumX650_1442.pdf.
5. https://www.juniper.net/documentation/en_US/junos/topics/task/configuration/multicast-limiting-group-joins-igmp.html.
6. https://www.manualslib.com/manual/232384/3com-5500g-Ei.html?page=247.
7. https://gtacknowledge.extremenetworks.com/articles/Q_A/What-is-the-maximum-number-of-IPv4-Multicast-groups-supported-on-Extreme-devices.
8. Billhartz T, Cain J, Farrey-Goudreau B et al. Performance and Resource Cost Comparisons for the CBT and PIM Multicast Routing Protocols, *IEEE Journal on Selected Areas in Communications*. 1997; 15(3): 304-315.
9. https://www.networkworld.com/article/2241579/multicast-group-capacity-extreme-comes-out-on-top.html.
10. Morse K, Zyda M. Multicast Grouping for Data Distribution Management. *Simulation Practice and Theory*. 2002; 9.
11. Wang J, Zheng T. A hybrid multicast-unicast assignment approach for data distribution management in HLA. *Simulation Modelling Practice and Theory*. 2014; 40(0): 39-63.

Streszczenie

W pracy przedstawiono problem komunikacji między federatami symulatora opartego o standard HLA. Omówiono zagadnienia sieciowe związane z komunikacją federatów oraz główne ograniczenia aspektów komunikacji sieciowej. Przedstawiono wybrane algorytmy wyboru strategii komunikacji wraz z opisem ich wad. Zaproponowano zmiany w doczasowych algorytmach, które mogą pozytywnie wpłynąć na wydajność komunikacji federatów.

Słowa kluczowe: symulacja, HLA, Data Distribution Management, Interest Management

Summary
Federate communication problem in HLA-based simulation

The paper presents communication problem between federated in HLA-based simulation. The networking background related to federates communication with consideration of limits of such communication was presented. Selected communication strategy algorithms and their weaknesses have been described. Improvements to those algorithms, which should improve communication performance, were proposed.

Keywords: Simulation, HLA, Data Distribution Service, Interest Management

DOI: 10.24427/978-83-66391-28-4_9

Metoda wyznaczania stanu agentów w symulacji wieloagentowej o zmiennej rozdzielczości

Dariusz PIERZCHAŁA [*]
Przemysław CZUBA [*]

1. Wprowadzenie

Świat w swojej naturze jest złożony z ogromnej liczby różnych współistniejących obiektów, które nieustannie ewoluują oraz stale oddziałują na siebie. Ich badanie za pomocą klasycznych metod, o wybiórczym zastosowaniu, często nie jest wystarczające z uwagi na szczegółowość i złożoność obiektów oraz konstruowanych dla nich modeli. W ramach poszukiwania nowych metod uformowała się gałąź inżynierii systemów, zwana systemami wieloagentowymi (ang. *multi-agent systems*, MAS). Jest to wiedza interdyscyplinarna, łącząca w sobie m.in. elementy systemów rozproszonych, sztucznej inteligencji, symulacji, teorii gier, a także nauk społecznych. Idea tychże systemów jest relatywnie prosta – składają się one z wielu agentów, które komunikują się ze sobą w ramach wspólnego środowiska (w tym są: cele, zasady, semantyka i ograniczenia). Autorzy Franklin i Gasser definiują agenta następująco [1]: „Autonomiczny agent jest systemem usytuowanym wewnątrz środowiska, które obserwuje i podejmuje w nim działania w celu osiągnięcia własnych celów". Bezpośrednią konsekwencją takiego stwierdzenia jest sformułowanie założenia o posiadaniu przez agenta zdolności do stawiania sobie zadań oraz dostosowywania swoich akcji w celu ich realizacji. Autonomiczna jednostka wykonuje swoje czynności w środowisku, w którym znajdują się inne agenty. Agent podczas deliberacji kolejnych działań musi brać pod uwagę również akcje innych agentów. Stąd też komunikacja oraz koordynacja działań pomiędzy nimi są niezbędne dla pomyślnej realizacji postawionych agentom zadań. To właśnie koordynacja działań w grupie agentów jest jednym z fundamentalnych problemów systemów wieloagentowych oraz stanowi jeden z aspektów rozważanych w niniejszym artykule.

Adekwatność w modelowaniu agentów i ich relacji sprawia, że symulacja złożonych systemów wieloagentowych może być bardzo wymagająca (co do zasobów i czasu realizacji). Każdy model jest pewną subiektywną abstrakcją wycinka rzeczywistości, a zatem modele mogą różnić się poziomem szczegółowości opisu struktury i dynamiki. Poziom szczegółowości zależy od trzech czynników: zakresu (systemu,

[*] Wojskowa Akademia Techniczna

domeny wejściowej i przetwarzanej informacji wyjściowej), rozdzielczości (dokładności, z którą przedstawiane są elementy systemu i ich zachowania) oraz perspektywy. Modelowanie wielorozdzielcze (ang. *multi-resolution modeling*, MRM) pozwala zróżnicować widzenie obiektu symulacyjnego i dostosować poziom rozdzielczości do bieżących oczekiwań. Jednostka (encja) o niskim poziomie rozdzielczości (wysokim stopniu agregacji) zwana jest LRE (ang. *low-resolution entity*) – reprezentuje często wiele jednostek zagregowanych w jednym obiekcie (np. batalion grupujący kompanie). Analogicznie, wysoki poziom rozdzielczości pociąga za sobą bardziej szczegółowy opis atrybutów oraz dynamiki zmian, zatem odpowiada niskiemu stopniowi agregacji – jednostki są reprezentowane indywidualnie (HRE – ang. *high-resolution entity*). Mimo że modele o wysokiej rozdzielczości odwzorowują systemy precyzyjniej, to modele zagregowane w dalszym ciągu są i będą implementowane w wielu systemach. Ma to związek z ograniczeniami mocy obliczeniowej i pamięci komputerów, a także uogólnionymi oczekiwaniami wobec modeli w określonych zastosowaniach. Złożone systemy adaptacyjne z zachowaniami emergentnymi (ang. *complex adaptive systems and emergent behaviors*) to przykładwystępowania pewnych zjawisk dopiero na poziomie makroskopowym (czyli niskiej rozdzielczości). Nie są one zrozumiałe przy zastosowaniu mikroskopowych praw rządzących pojedynczymi obiektami, ponieważ zachowanie emergentne pojawia się dopiero w analizie grupy prostych jednostek jako jednego agregatu. Jego uogólnione zachowanie jest bardziej złożone niż zachowania składowych obiektów, a w przypadku emergencji związanej z różnicami poziomów powodem może być uporządkowanie i wzmocnienie specyficznych oddziaływań między jednostkami, skutkujące synergicznymi efektami w całej grupie.

Jednostka, którą można obserwować na wielu różnych poziomach rozdzielczości nazywana jest encją wielorozdzielczą (ang. *Multi-resolution Entity,* MRE). W momencie interakcji między jednostkami na różnych poziomach rozdzielczości dochodzi z reguły do problemu spójności stanów – obiekty nie mogą poprawnie współpracować, gdy posiadają różne zestawy atrybutów, a ich dynamikę odwzorowują różne procesy. Typowym rozwiązaniem jest adaptacyjne dostosowanie rozdzielczości w taki sposób, aby interakcja zachodziła pomiędzy obiektami na tym samym poziomie. Na przestrzeni ostatnich 30 lat powstało wiele rozwiązań tego problemu – w opracowaniu opieramy się na metodzie cross-resolution (CRM). Proponujemy wykorzystanie technik wieloagentowych w symulacji o zmiennej rozdzielczości, gdzie w agregacji oraz deagregacji stanu agenta stosowane są algorytmy konsensusu [2] oraz sterowania formacją [3].

W kolejnej części opracowania zamieszczone jest zwięzłe wprowadzenie do sieci wieloagentowych, które są istotne dla prezentowanych prac, a także opis propozycji metody agregacji i deagregacji stanu agenta. Kolejna część zawiera charakterystykę pakietu symulacyjnego, który został wykorzystany do autorskiej implementacji metody. Finalnie opisano studium przypadku i zaprezentowano wyniki eksperymentalne.

2. Podstawowe pojęcia i algorytmy sieci wieloagentowych

2.1. Sieć wieloagentowa

Sieć wieloagentową (ang. *multiagent network*) możemy widzieć jako zbiór dynamicznych jednostek, które współdziałają przez wymianę sygnałów dla skoordynowania własnego zachowania oraz osiągnięcia wspólnego celu [3]. Podstawowym założeniem dla sieci jest fakt, iż jej struktura oraz cechy wpływają na dynamiczne właściwości systemu. Rozproszone sieci wieloagentowe (np. system rozproszonych robotów) wniosły do teorii sieci nowe problemy związane z ich analizą. Agenty w takich sieciach powinny współdziałać w sposób skoordynowany, aby osiągnąć zbiorowy synergiczny cel, posiadając do dyspozycji ograniczone zasoby obliczeniowe oraz możliwości komunikacji i percepcji.

W sieciach wieloagentowych zakłada się istnienie:

- autonomicznych i dynamicznych jednostek (agentów), które posiadają umiejętności podejmowania decyzji oraz wymiany informacji z sąsiadami;
- struktury wymiany informacji, którą modeluje się przy użyciu teorii grafów – wierzchołki reprezentują jednostki z ograniczonymi zasobami, a gałęzie wirtualne encje kodujące przepływ informacji pomiędzy wierzchołkami.

Poniżej przedstawione zostały wybrane pojęcia sieci wieloagentowych:

- $x_i(t)$, $i \in N$ – stan wierzchołka (agenta) *i-tego* w chwili *t*;
- N_i – dla grafu nieskierowanego: zbiór wszystkich gałęzi incydentnych (sąsiadujących) z wierzchołkiem (agentem) *i-tym*, natomiast dla grafu skierowanego zbiór jego poprzedników, czyli łuków wchodzących do wierzchołka *i-tego*;
- $I_i(t) = \{x_j(t) | \, j \in N_i$ – opis wiedzy agenta *i-tego* o sąsiadach w chwili *t*; jest to zbiór stanów agentów będących sąsiadami (w przypadku grafu nieskierowanego) bądź poprzednikami (graf skierowany);
- $x_i(t+1) = F_i(x_i(t), I_i(t))$ – reguła wyznaczenia stanu agenta *i-tego* w kolejnej chwili, oparta na jego aktualnym stanie oraz wiedzy o sąsiadach.

Sieci wieloagentowe wykorzystywane są w szczególności do rozwiązywania problemów związanych z osiąganiem konsensusu oraz ustalaniem formacji agentów.

2.2. Algorytm konsensusu

W systemach wieloagentowych algorytm konsensusu wpisuje się w klasę problemów związanych ze sterowaniem kooperacyjnym (ang. *cooperative control for multiagent systems*). W przypadku sieci wieloagentowych konsensus oznacza osiągnięcie zgody (ang. *agreement*) co do szczególnej wartości, która jest zależna od stanu wszystkich

agentów w sieci. Ogólna zasada działania algorytmu konsensusu (zwanego także protokołem konsensusu) polega na określeniu i zastosowaniu reguł interakcji, opisujących sposób wymiany informacji pomiędzy agentem i jego sąsiadami w sieci, w wyniku których uzgodniona zostanie wspólna wartość. Podczas procesu uzgadniania stany poszczególnych agentów ewoluują w czasie. Mówi się, że system osiągnął konsensus, gdy wartości dla uzgadnianego stanu *u* każdego z agentów w sieci są równe. Reguły w procesie uzgadniania wykorzystują funkcje definiowane adekwatnie do specyfiki problemu (np. wartość minimalna, maksymalna, średnia itp.)

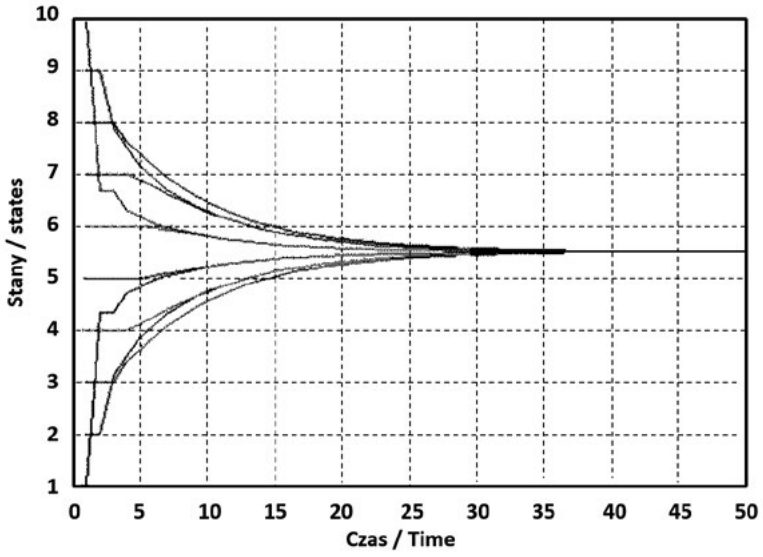
Proponowana metoda agregacji stanów agentów jest oparta na poniższym protokole konsensusu [4], z założeniem dyskretnego czasu oraz topologii stałej w czasie:

$$x_i(t+1) = \frac{1}{N_i + 1}\left(x_i(t) + \sum_{j \in N_i} x_j(t)\right)$$

Z protokołu wynika, że stan agenta (wartość uzgadniana) w każdej następnej chwili jest średnią arytmetyczną z jego stanu oraz stanów sąsiadów. Zaznaczymy, że w przypadku dynamicznych topologii zbiór sąsiadów agenta jest zmienny w czasie [3].

Poniższa granica przedstawia spodziewany wynik algorytmu. Powinien on być równy średniej arytmetycznej początkowych stanów agentów:

$$\lim_{t \to \infty} x_i(t) = \frac{1}{N} \sum_{j \in N} x_j(0), i \in N$$



RYS. 1. Ewolucja stanu agentów podczas działania algorytmu konsensusu
FIG. 1. Agents' states evolution according to consensus algorithm

ŹRÓDŁO: [3].
SOURCE: [3].

Na rysunku 1 przedstawiono przykład ewolucji stanów agentów podczas uzgadniania wspólnej wartości dla stanu encji o niższym poziomie rozdzielczości.

2.3. Sterowanie formacją

Formację można zdefiniować jako wzorzec geometryczny realizowany przez grupę składającą się z wielu agentów. Idea formacji wywodzi się z badań inspirowanych biologią – np. ptaki, używając ustalonej formacji, minimalizują utratę sił podczas lotu. Sterowanie formacją (ang. *formation control*) jest jednym z istotniejszych problemów poruszanych w zagadnieniach sterowania oraz koordynacji agentów w systemach wieloagentowych lub wielorobotowych.

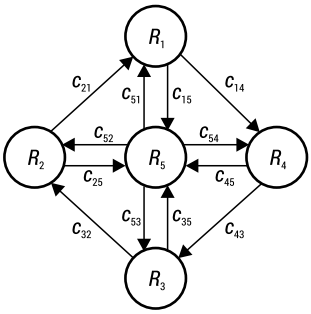
Ideą rozważanej metody jest poruszanie się agentów w ich środowisku tak, aby pozostał utrzymany kształt (wzorzec) formacji. Jednostki w formacji nie poszukują konsensusu co do swoich stanów w całości, lecz co do ich relatywnej pozycji względem reszty agentów. Wzorując się na regułach koordynacji obserwowanych w naturze, wszystkie agenty programowe muszą przestrzegać lokalnych zasad opartych na częściowej wiedzy o pozycji określonych członków grupy. Przykładem w systemie wielorobotowym może być zadanie eksploracji, gdy roboty poruszają się w konkretnej formacji tak, aby maksymalizować przeszukiwany obszar.

W sterowaniu formacją wyróżnia się zasadniczo dwa podejścia:

- pierwsze, gdzie agenty działają według prostych reakcyjnych reguł zachowania, utrzymując dystans od sąsiadów bez zachowania konkretnej pozycji w formacji oraz jej kształtu – schemat takiego działania jest charakterystyczny dla metod inteligencji roju (ang. *swarm intelligence*), a przykładem realizacji może być model boidów Reynoldsa [5];
- drugie oparte jest na określonej odgórnie topologii sieci komunikacji pomiędzy agentami, nazywanej grafem formacji (ang. *formation graph, FG*); wierzchołek w grafie reprezentuje pozycję agenta, a gałęzie – możliwe kanały wymiany informacji pomiędzy parami jednostek (jedno lub dwukierunkowe); rysunek 2 przedstawia graficzny przykład takiego grafu formacji.

Poprawnie zdefiniowany graf formacji musi być spójny, tzn. bez wyizolowanych wierzchołków. Dla każdej gałęzi w grafie formacji określony jest wektor pożądanej relatywnej pozycji pomiędzy parą wierzchołków (pozycjami agentów).

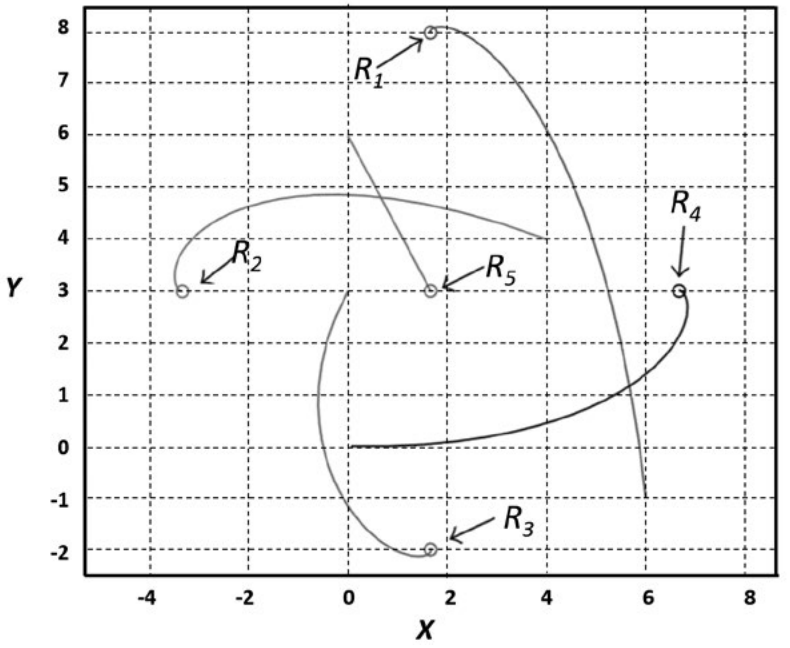
Większość podejść do sterowania formacją zakłada ciągły upływ czasu. Przykładami realizacji są: konfiguracja konwojów w pracy Belkhouche'a [6] czy formacja cyklicznego pościgu Francisa, Broucke'a i Lina [7]. Z drugiej strony, sterowanie formacją w oparciu o czas dyskretny jest stosowane w algorytmie konsensusu, co konceptualnie różni się od podejścia z czasem ciągłym (dodane wektory relatywnych pozycji pomiędzy agentami).



RYS. 2. Przykład grafu formacji
FIG. 2. Example of formation graph

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

W niniejszym opracowaniu do realizacji metody deagregacji stanu agentów użyto algorytmu sterowania formacją, który jest dyskretny w czasie oraz zakłada stałą topologię [8]. Jego realizacja opiera się na grafie formacji. Stan początkowy agenta jest równy wartości stanu jednostki (encji) o niższej rozdzielczości.



RYS. 3. Przykład grafu formacji
FIG. 3. Example of formation graph

ŹRÓDŁO: [8].
SOURCE: [8].

Niech N_i będzie zbiorem poprzedników agenta R_i, czyli zbiorem agentów, które wykrywają jego pozycję. Niech c_{ij} dla wszystkich $j \in N_i$ oznacza niezależny od czasu wektor pożądanych relatywnych pozycji R_i w zależności od pozycji R_j. Oznacza on odległości, jakie powinny być utrzymywane w konkretnej formacji pomiędzy parami agentów. Stąd pożądaną relatywną pozycję dla każdego R_i w formacji opisujemy przez:

$$z_i^*(t) = \frac{1}{N_i} \sum_{j \in N_i} \left(z_j + c_{ji} \right), i \in N$$

Zatem pożądane relatywne pozycje agentów możemy interpretować jako kombinację pożądanych pozycji z_i względem pozycji wszystkich elementów N_i (poprzedników danego agenta).

Strategia kontroli formacji dla deagregacji stanu agenta posiada następującą postać:

$$u_i(t) = -k\left(z_i(t) - z_i^*(t) \right), i \in N$$

gdzie $k > 1$ jest parametrem przyrostu. Na rysunku 3 przedstawiono wyniki symulacji dla grafu formacji z rysunku 2.

3. Symulacja dyskretna w pakiecie DisSim

W prowadzonych pracach przyjęto, że symulacja komputerowa to ilościowa i jakościowa metoda modelowania w języku formalnym oraz odwzorowania w programie komputerowym strukturalnych i behawioralnych cech systemów (rzeczywistych lub projektowanych), umożliwiająca eksperymentowanie z modelem (zamiast z systemem) i obserwowanie w nim procesów zachodzących w symulacyjnym czasie. W praktyce modele symulacyjne implementowane są z wykorzystaniem bibliotek programowych o różnych możliwościach (np. w zakresie zarządzania czasem i zdarzeniami symulacyjnymi), a zatem o różnych wymaganiach sprzętowych. Dobór właściwego języka programowania i jego symulacyjnych rozszerzeń może być pokierowany bardzo zróżnicowanymi kryteriami – od znajomości języka, przez koszty czasowo-finansowe, aż po możliwość wykonania symulacji w wybranym języku. Oprócz „ciężkich" pakietów symulacyjnych można znaleźć tzw. lekkie rozwiązania, które budowane są od podstaw i tym samym krojone na potrzeby konkretnego zastosowania. W przypadku badań, będących tematem niniejszej pracy, istotne jest, aby czas w symulacji upływał w sposób dyskretny lub quasi-ciągły – ma to bezpośredni związek z algorytmami konsensusu i sterowania formacją.

Autorska propozycją, wychodzącą naprzeciw takim potrzebom, jest pakiet do symulacji dyskretnej DisSim. Agenty wchodzące w skład systemu, wraz z ich cechami oraz relacjami łączącymi je w związki, są odwzorowane w zbiorze obiektów O [9, 10]:

- $O = \{o = < id, c >\}, c \in C°, id \in \mathrm{N}$ – zbiór symulowanych obiektów klasy c identyfikowanych przez id o wartości niepowtarzalnej w zbiorze obiektów;
- $C°$ – niepusty zbiór klas modelowanych obiektów.

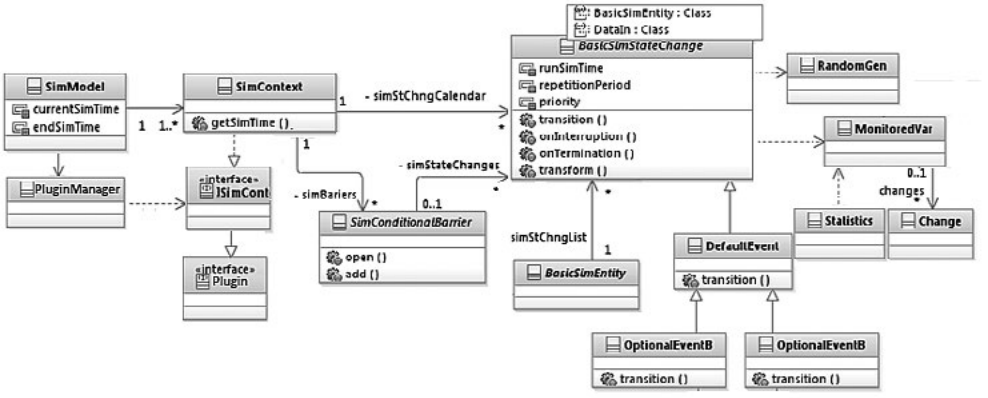
Do każdej istotnej dla celu modelowania, mierzalnej cechy agenta należy przypisać atrybut opisany dziedziną dopuszczalnych wartości:

- A_c – niepusty zbiór atrybutów określonych dla klasy obiektów $c \in C°$;
- V_a^c – zbiór dopuszczalnych wartości atrybutu $a \in A_c$ klasy obiektów $c \in C°$.

W praktyce zbiory $C°$ oraz A_c przedstawia się jako zbiory numerów odpowiednio modelowanych klas obiektów i ich atrybutów.

W symulacji dynamicznej niezbędne jest zdefiniowanie metod i algorytmów programowych wyznaczania kolejnych stanów agenta. W każdym punkcie t czasu symulacyjnego każda składowa stanu systemu, a tym samym wyróżnione cechy agentów, widziana będzie jako czwórka uporządkowana: $s_{o,a} = <o, a, v, t>$. Zatem stan modelowanego systemu $S(t) = \{<o, a, v, t>\}$, $o \in O$ będzie zbiorem utworzonym przez atrybuty wszystkich obiektów (agentów) istniejących w chwili symulacyjnej t. W symulacji dynamicznej wartości $v \in V_a^c$ atrybutów $a \in A_c$ wyznaczane są przez funkcje zmiany stanu, tworząc tzw. zdarzenia.

W *podejściu zorientowanym na zdarzenia* pod pojęciem zdarzenia e ze skończonego zbioru zdarzeń E rozumiana jest planowa zmiana stanu obiektu w określonym punkcie czasu symulacyjnego: $e = <t, f_e^S(t)>$, $t \in \mathrm{T}$. Funkcja zmiany stanu f_e^S: $SxT \rightarrow SxT$ wyznacza stan, w jakim znajdzie się system w chwili t po zajściu zdarzenia e. W metodach symulacji dyskretnej krokowej oraz zdarzeniowej przyjmuje się następujące uproszczenie modelowe – stan systemu nie ulega zmianie do czasu realizacji kolejnego zdarzenia, czyli w przedziale $[t_i, t_{i+1}) \subset T$. Jeżeli zdarzenia występują w odstępach czasu pomijalnych z punktu widzenia uproszczeń modelowych, możemy traktować upływ czasu i symulację jako quasi-ciągłą.



RYS. 4. Podstawowe klasy pakietu DisSim
FIG. 4. Basic classes of DisSim package

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Pakiet DisSim zrealizowano w języku Java. Stanowi bazową warstwę programową symulatora realizującego w praktyce proponowane modele i metody. Podstawowe znaczenie mają klasy zdarzenia (*BasicSimStateChange*) oraz obiektu symulacyjnego (*BasicSimEntity*). Inne klasy to m.in. klasy odpowiadające za generowanie liczb pseudolosowych (*RandomGen*), monitorowanie zmiennych (*MonitoredVar*), oszacowania statystyczne (*Statistics*) czy przesyłanie komunikatów (*EventBroker*). Z ich pomocą opracowano drugą warstwę programową symulatora, odpowiedzialną za odwzorowanie agentów oraz wielorozdzielczości. Do implementacji jednostki (encji) wielorozdzielczej wykorzystano klasę obiektu symulacyjnego *BasicSimEntity* – oparta na niej klasa *ResolutionLevel* jest abstrakcją reprezentującą obiekt na określonym poziomie rozdzielczości. Dwa interfejsy – *IAggregation* i *IDeaggregation* – są zapowiedzią metod przejść pomiędzy poziomami szczegółowości, czyli agregacji i deagregacji Dzięki zastosowaniu „wstrzykiwania zależności" implementacje metod agregacji i deagregacji mogą być definiowane na zewnątrz klasy *ResolutionLevel*, a zatem mogą być modyfikowane w trakcie symulacji. Grupując obiekty *ResolutionLevel* w ramach jednej klasy, otrzymujemy jednostkę Multiple-Resolution Entity (MRE) o wielu poziomach rozdzielczości. Zmiana poziomu rozdzielczości jest publikowana jako zdarzenia klas: *AggregationEvent*, *DeaggregationEvent*.

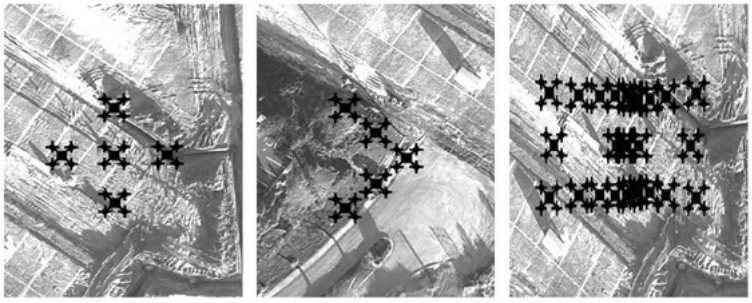
Klasa *BasicSimEntity* jest bazową również dla *BasicAgent* – klasy agenta programowego. Jedną z cech agenta jest możliwość reagowania na zmiany występujące w otaczającym go środowisku. Zatem, każdy taki obiekt implementuje interfejs IEventSubscriber, dzięki czemu otrzymuje dostęp do funkcji pakietu DisSim dotyczących subskrypcji zdarzeń. Ponadto, wykorzystany jest model filtrowania z pakietu *DisSim*, ograniczający przesyłanie komunikatów między publikującymi a subskrybentami. Klasa *BasicAgent* zawiera atrybuty identyfikujące agenta (*id*) i pozycjonujące agenta w przestrzeni (*position*). Model komunikacji (interakcji) odwzorowany jest w klasach pakietu *Network*, opartego na bibliotece *JGraphT*. Główna klasa *Network* reprezentuje sieć wieloagentową, złożoną z obiektu klasy *DirectedWeightedMultigraph* (z biblioteki *JGraphT*), wierzchołków sieci w postaci obiektów klasy dziedziczącej z *BasicAgent* oraz połączeń pomiędzy parami agentów jako klasy *Link*. W obiekcie klasy *Link* atrybuty odziedziczone z klasy *DefaultWeightedEdge* (biblioteki *JGraphT*) opisują wierzchołki (agentów) źródłowych i docelowych połączenia, wagę wierzchołka oraz wektor *Point2D*. Docelowo klasa *Network* zastosowana została do implementacji klasy *FormationGraph*, która reprezentuje możliwe kierunki interakcji pomiędzy agentami oraz wektory pożądanych relatywnych pozycji *PositonVector*. Wektory pozycji są definiowane przez agentów (sourceAgentId, targetAgentId) będących w połączeniu, a także wektor pożądanej relatywnej pozycji dla połączenia.

Przyjęta koncepcja symulacji oraz jej realizacja w opisanych warstwach pakietów i klas programowych języka Java pozwalają na definiowanie i implementację modeli symulacyjnych z rozdzielczością adekwatną do modelowanego problemu oraz z upływem czasu dyskretnym lub quasi-ciągłym.

4. Eksperyment symulacyjny z grupą statków powietrznych BSP

Model symulacyjny wykorzystany w eksperymentach odwzorowuje grupę (drużynę) bezzałogowych statków powietrznych (BSP) z jej strukturą i dynamiką. Zaimplementowany został z wykorzystaniem opisanego w poprzednim punkcie pakietu DisSim, rozszerzonego o warstwę grafiki z biblioteką JavaFX. Agentem programowym jest statek BSP, a drużyna to zagregowany obiekt w ramach MRE.

Przyjmiemy dla potrzeb symulacji następujące założenia dotyczące programowej realizacji modelu i przebiegu eksperymentu. W chwili startu symulacji drużyny dronów są prezentowane na poziomie zagregowanym i rozpoczynają od zadania patrolowania zadanego obszaru. W momencie, gdy drużyna dronów otrzyma sygnał, że na obszarze pojawił się obiekt wymagający ich działania (dla drużyny poszukującej – ratunek, dla drużyny atakującej – eliminacja), zagregowane statki BSP lecą do miejsca zdarzenia. Po dotarciu wykonana zostaje deagregacja obiektu MRE w celu wykonania zadania przez poszczególne agenty (pojedyncze BSP). Po zakończeniu działania drużyna agreguje się do obiektu grupowego i powraca do patrolowania. Kolejny rysunek (5) przedstawia różne formacje zaimplementowane w testach.



RYS. 5. Formacje statków BSP na wysokim poziomie rozdzielczości: formacja „poszukiwawcza", formacja „atakująca" i formacja „żółwia" bez punktów stabilizacyjnych

FIG. 5. High-resolution formations of BSP ships: "search", "attacking", "turtle" without stabilization points

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Badania testowe zostały przeprowadzone na następującej konfiguracji sprzętowej: Intel Core i5-3230M 2.60GHz, 12GB RAM, system operacyjny Kubuntu 64-bit. Podczas badań zmierzone zostały czasy wykonania metod agregacji i deagregacji drużyny dronów w celu określenia ich zależności od topologii sieci wieloagentowej. Pomiary wykonano przy użyciu dostępnej w języku Java metody *System.currentTimeMillis*().

Warianty eksperymentów różniły się następującymi parametrami:

- trybem symulacji – z aktywną wizualizacją z krokiem czasu symulacyjnego 0.5 sekundy dla wyznaczania kolejnych pozycji agentów oraz w trybie ASAP (najszybszy możliwy czas wykonania symulacji);
- rodzajem formacji – badania zostały przeprowadzone dla czterech różnych typów formacji, różniących się zasadniczo topologią (rys. 4):
  - formacja atakująca składająca się z pięciu wierzchołków,
  - formacja poszukiwawcza składająca się z pięciu wierzchołków,
  - formacja konwojowania składająca się z pięciu wierzchołków,
  - formacja „żółwia" składająca się z piętnastu wierzchołków.

Czas był mierzony od momentu rozpoczęcia procesu agregacji (deagregacji) aż do momentu jego zakończenia. Wyniki pomiarów zostały przedstawione w tabeli 1.

TAB. 1. Wyniki symulacji
TAB. 1. Simulation results

TRYB Z AKTYWNĄ WIZUALIZACJĄ					
Formacja atakująca					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	503	500		511	500
Czas deagregacji [ms]	6837	7101		7108	7106
Formacja poszukiwawcza					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	1005	1000		699	700
Czas deagregacji [ms]	4825	4800		4800	4799
Formacja konwojowania					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	506	502		504	502
Czas deagregacji [ms]	24739	28415		28414	29212

TRYB Z AKTYWNĄ WIZUALIZACJĄ					
Formacja „żółwia”					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	1528	1601		1599	1603
Czas deagregacji [ms]	32539	35524		35703	43869
TRYB ASAP					
Formacja atakująca					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	4	2		11	14
Czas deagregacji [ms]	92	67		69	152
Formacja poszukiwawcza					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	9	9		23	22
Czas deagregacji [ms]	43	79		104	64
Formacja konwojowania					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	11	7		8	7
Czas deagregacji [ms]	272	280		302	241
Formacja „żółwia”					
Lp.	1.	2.	…	9.	10.
Czas agregacji [ms]	55	22		45	34
Czas deagregacji [ms]	641	645		565	581

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Wykonany eksperyment pokazał, iż w obu trybach (z aktywną wizualizacją oraz ASAP) dla formacji „atakująca" oraz „poszukiwawcza" zarejestrowano różne czasy, pomimo tej samej liczby wierzchołków w sieci. O ile czasy dla agregacji (algorytm konsensusu) są wyraźnie zbliżone, to dla metody deagregacji (sterowanie formacją) różnią się nawet o tysiące milisekund w obu trybach. Prowadzi to do wniosku, że topologia sieci ma wpływ na czas wykonania deagregacji agentów. W zależności od połączeń pomiędzy agentami (tu statkami BSP) ustalanie formacji może być mniej lub bardziej złożone czasowo. Czas wykonania rośnie także z rozmiarem sieci. Jest to przewidywalny wniosek i potwierdza zależność, iż zwiększanie liczby wierzchołków wiąże się z większą ilością obliczeń.

5. Podsumowanie

W badaniach zaproponowano wykorzystanie technik wieloagentowych w symulacji o zmiennej rozdzielczości, przy czym w agregacji oraz deagregacji stanu agenta zastosowano algorytmy konsensusu oraz sterowania formacją. Podejście wieloagentowe oraz rozdzielcze w modelowaniu systemów są ze sobą połączone w sposób naturalny.

Przyjęta koncepcja symulacji oraz jej realizacja w postaci biblioteki DisSim z dziedzinowymi rozszerzeniami do implementacji modeli MRE pozwala na odwzorowanie w symulacji obiektów z rozdzielczością adekwatną do modelowanego problemu oraz z upływem czasu dyskretnym lub quasi-ciągłym.

Proponowane multidyscyplinarne podejście wydaje się bardzo obiecujące w symulacji wielorozdzielczej. Zaadaptowane algorytmy konsensu i sterowania formacją ograniczają konieczność definiowania znacznie bardziej złożonych algorytmów na potrzeby agregacji i deagregacji w tej klasie problemów. Następnym krokiem może być wykorzystanie technik uczenia maszynowego, co mogłyby poprawić efekty szukania przez agentów optymalnych formacji w zależności od warunków otoczenia.

Literatura

1. Franklin S, Gasser A. Is it an agent, or just a program?: A taxonomy for autonomous agents. In: Muller J, Wooldridge MJ, Jennings NR. eds *Intelligent Agents III. Agent Theories, Architectures, and Languages*. Budapest: Springer Verlag; 1997; 21-35.
2. Saber RO, Murray RM. Consensus protocols for networks of dynamic agents. *American Control Conference Proceedings*. 2003; 951-956.
3. Mesbahi M, Egerstedt M. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press; 2010.
4. Zhipu J. *Consensus Problem and Algorithms*. CDS 270-2: Lecture 8-1; 2006.

5. Reynolds CW. Flocks, Herds, Schools. *A Distributed Behavioral Model*. ACM SIGGRAPH; 1987.
6. Belkhouche F, Belkhouche B. Modeling and controlling a robotic convoy using guidance laws strategies. *IEEE Transactions on Systems, Man, and Cybernetics B*. 2005; vol. 35, no. 4: 813-825.
7. Francis B, Broucke M, Lin Z. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*. 2004; vol. 49, no. 4: 622-629.
8. Hernandez-Martinez EG, Flores-Godoy JJ, Fernandez-Anaya G. *Decentralized Discrete-Time Formation Control for Multirobot Systems*. Universidad Iberoamericana; 2013.
9. Pierzchała D., *Symulacja komputerowa – od procedury do chmury*, w monografii ISBN/ISSN: 978-83-7938-038-1. Warszawa; 2014; 104-118.
10. Dyk M, Najgebauer A, Pierzchała D. Agent-based M&S of smart sensors for knowledge acquisition inside the Internet of Things and sensor networks. *Int. Inf. and Database Syst*., LNCS, 9012, Subseries: LNAI, XXXVI; 212-223; 2015.

Streszczenie

W opracowaniu zaproponowano wieloagentowe podejście do wyznaczania stanu agenta w symulacji wielorozdzielczej (o zmiennej rozdzielczości) i wieloagentowej. Dwie kluczowe metody zastosowane do realizacji procesu agregacji i deagregacji stanów to algorytm konsensusu i kontroli formacji. Idea koordynacji działań wielu agentów wyłoniła się z obserwacji oraz symulacji zbiorowych zachowań żywych istot. Algorytmy konsensusu są powszechnie stosowane w przypadku problemów sterowania kooperacyjnego w systemach wieloagentowych (konsensus oznacza osiągnięcie zgody na temat szczególnej wartości, która jest zależna od stanu wszystkich agentów w sieci). Kontrola formacji jest natomiast najpopularniejszym algorytmem w problemie koordynacji ruchu w systemach wielorobotowych, gdzie musi być spełniony warunek utrzymania predefiniowanego kształtu geometrycznego formacji.

Przedstawione w pracy podejście pokazuje, że metody wielodyscyplinarne wydają się bardzo obiecujące w symulacji wielorozdzielczej. Algorytmy konsensu i kontroli formacji eliminują konieczność definiowania znacznie bardziej złożonych algorytmów na potrzeby agregacji i deagregacji.

Słowa kluczowe: wielorozdzielcza symulacja, symulacja wieloagentowa, system wieloagentowy, sterowanie formacją grupy

Summary
The method of state estimation in multiagent multiresolution simulation

The paper proposes the multiagent techniques for estimation of agent's state in the multiresolution multiagent simulation. The key methods we have used for state aggregation and disaggregation are: consensus algorithm and formation control. The idea of the coordination of multiple agents has emerged from both observation and simulation of a collective behaviour of biological entities.

The consensus algorithms are commonly used for the cooperative control problems in the multiagent systems, whilst the formation control is the most popular and fundamental motion coordination problem in the multiagent systems, where agents converge to predefined geometric shapes.

The presented approach shows that multiagent methods seem to be very promising in multiresolution simulation. Consensus and formation control algorithms remove necessity to specify the much more complex algorithms for the aggregation and disaggregation needs.

Keywords: multiresolution multiagent simulation, multiagent system, formation control

DOI: 10.24427/978-83-66391-28-4_10

Symulator systemów klasy SOA

Adrian P. WOŹNIAK[*]

1. Architektura zorientowana na usługi

Architektura zorientowana na usługi (ang. *Service Oriented Architecture*, SOA) to sposób wytwarzania oprogramowania mający na celu w optymalny sposób wspierać organizację. Składa się na to wiele cech, z których najważniejszymi są czas i koszty wprowadzania zmian. Dzięki stosowaniu usług można zapewnić niezależność w rozwoju poszczególnych systemów i rozwijać je równolegle. Pozwala to na wprowadzanie zmian w organizacji w krótkim czasie, co staje się coraz bardziej istotne we współczesnym, szybko zmieniającym się świecie. Od 15 lat podejście to zyskuje na popularności i jest coraz szerzej wdrażane w przedsiębiorstwach. Współcześnie koncepcja ta jest rozwijana z wykorzystaniem coraz to nowszych technologii. Skutkuje to pojawieniem się nowych koncepcji rozszerzających SOA, takich jak np. tzw. mikrousługi (ang. *microservices*), gdzie nowe systemy tworzone są z niewielkich komponentów odpowiedzialnych za wydzielony i dobrze wyspecyfikowany obszar biznesowy.

Centralnym pojęciem w SOA jest usługa, która jest niezależną funkcją systemu, dającą wartość z punktu widzenia procesu biznesowego. Każda usługa może być więc krokiem w procesie biznesowym. Najczęściej są one udostępniane poprzez sieć w formie tzw. Web Services. Usługi dostarczane są przez komponenty, czyli oprogramowanie, które jest względem siebie niezależne. Oznacza to, że każdy komponent może zostać wytworzony w innej technologii i działać na innym serwerze, w innym środowisku uruchomieniowym. Jest to poprawne z punktu widzenia SOA, dopóki usługi udostępniane są w uzgodnionej wcześniej formie. Elastyczność SOA pozwala na to, aby z jednej strony wytwarzać systemy w formie np. tzw. mikrousług, a z drugiej strony modyfikować istniejące stare systemy tak, aby udostępniać ich funkcje w formie usług. Powoduje to rosnącą liczbę stosowanych technologii, środowisk uruchomieniowych (rozumianych jako oprogramowanie potrzebne do uruchomienia komponentu) i serwerów. W celu realizacji procesu biznesowego potrzebne jest wywołanie wielu usług udostępnianych przez liczne komponenty. Oznacza to, że realizację procesu biznesowego angażowanych może być wiele serwerów komunikujących się poprzez sieć.

[*] Wojskowa Akademia Techniczna

Istotne jest zatem takie wdrożenie komponentów na serwerach, aby proces biznesowy realizowany był jak najlepiej z punktu widzenia organizacji. Poprzez „jak najlepiej" rozumiemy tutaj optymalizację według czterech kryteriów:

- czasu i wariancji procesu biznesowego,
- zużycia procesora i pamięci RAM na rzecz usług względem wszystkich zarezerwowanych zasobów.

Zastosowanie pierwszych dwóch kryteriów wydaje się być intuicyjne – z punktu widzenia organizacji wartościowe jest, aby proces biznesowy realizowany był w czasie jak najkrótszym i żeby czas ten był jak najbardziej przewidywalny. Minimalizacja według pozostałych dwóch kryteriów ma na celu jak najlepsze wykorzystanie zasobów organizacji. Zastosowanie wielu serwerów o dużej mocy obliczeniowej może dać lepsze wyniki pod względem pierwszych dwóch kryteriów, ale niekoniecznie musi być optymalne dla firmy, ponieważ może się okazać, że serwery te będą w niewielkim stopniu wykorzystywane, a takie rozwiązanie byłoby nieekonomiczne.

Podjęte próby optymalizacji w obszarze SOA skupiały się głównie na procesie tzw. Service Selection, a więc na algorytmie wyboru usługi w przypadku, gdy wiele instancji komponentu udostępnia tę samą usługę i należy wybrać ten, który ją zrealizuje. Wykorzystano do tego wiele koncepcji i algorytmów. W [1], [2], [3], [4] i [5] można znaleźć przykłady metod Service Selection opartych o algorytm genetyczny. Metoda polegająca na przeszukiwaniu drzew binarnych została opisana w [6]. Znacząco mniej popularnym zagadnieniem jest optymalizacja przydziału komponentów do serwerów. Tekst [7] przedstawia metodę optymalizującą przydział pod względem kosztowym przy ograniczeniach, jakie powoduje SLA (ang. *Service Level Agreement*). W artykule [8] można znaleźć metodę optymalizującą dostępność usług. Natomiast w [9] połączone są aspekty optymalizacji Service Selection (optymalizacja krótkoterminowa) z długoterminowo optymalnym przydziałem zasobów do serwerów. W literaturze można także znaleźć metody harmonogramowania usług. Definiowane jest ono jako wybór kolejności realizacji usług w kolejce komponentu. FIFO jest najczęściej stosowanym, choć niekoniecznie optymalnym, sposobem organizacji kolejki. Propozycję metody harmonogramowania usług można znaleźć w [10]. Jej działanie polega na znalezieniu usług będących na tzw. ścieżce krytycznej procesu i priorytetyzowaniu realizacji właśnie ich. W [11] natomiast można znaleźć bardziej rozbudowaną metodę, która optymalizuje kolejność na dwóch poziomach:

- globalnym, którym jest maksymalizacja prawdopodobieństwa spełnienia ograniczeń jakości usług (QoS),
- lokalnym, którym jest realizacja jednej z czterech polityk: najpierw usługi o największej wartości dodanej, najpierw usługi o najkrótszym wymaganym czasie realizacji (wynikającym z QoS), najpierw usługi o największej proporcji poprzednich dwóch kryteriów, najpierw usługi wybrane metodą Lawlera.

Jak dotąd nie zostały opublikowane metody optymalizujące przydział komponentów do serwerów w architekturze SOA, uwzględniające jednocześnie opisane wcześniej złożone działanie takich systemów, niezawodność serwerów oraz algorytmy wyboru usług i kolejności ich realizacji w kolejkach. Do przygotowania takiego optymalizatora proponowany jest algorytm genetyczny, którego celem jest znalezienie optymalnego przydziału komponentów do serwerów przy zadanych algorytmach Service Selection oraz wybór kolejności realizacji usług w kolejkach. Najtrudniejszym elementem takiego optymalizatora jest ocena rozwiązania. Do oceny wykorzystany został symulator, którego celem jest pomiar wymienionych wcześniej czterech kryteriów rozwiązania.

2. Środowisko symulacyjne

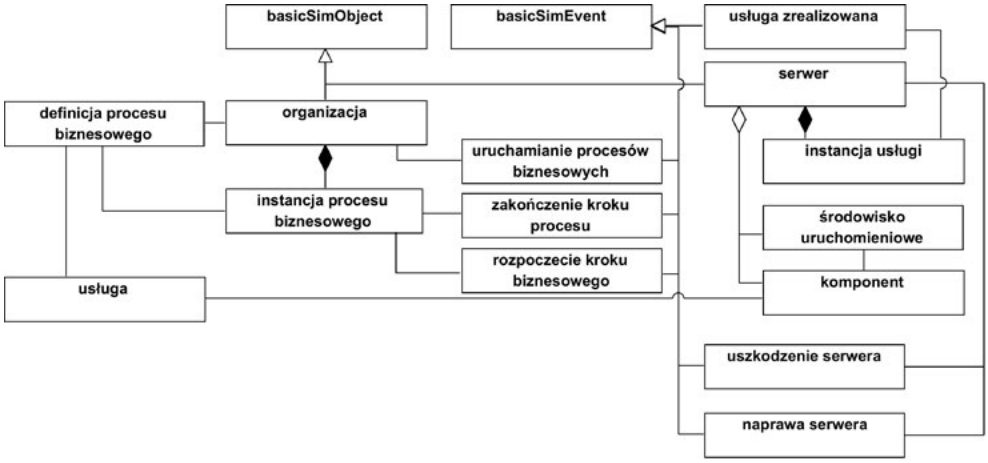
Symulator systemów klasy SOA został zaimplementowany z wykorzystaniem dwóch środowisk symulacyjnych: DESKit oraz DISSim. Obydwa środowiska zostały napisane w języku JAVA oraz wspierają symulację dyskretną zdarzeniową, która została tu zastosowana. Występuje jednak między nimi znacząca różnica koncepcyjna. W środowisku DESKit wykorzystuje się dwie najważniejsze klasy: BasicSimObject oraz BasicSimActivity. Pierwsza z nich reprezentuje obiekty symulacyjne, takie jak np. serwer czy organizacja, natomiast druga działania tych obiektów – np.: instancję procesu biznesowego i realizację usług przez serwer. Najważniejszą cechą tego środowiska jest to, że każda aktywność jest reprezentowana jako osobny wątek. Symulator oparty o to środowisko wznawia i zatrzymuje wątki zgodnie z kolejnością zapisaną w tzw. Pending List. W środowisku DISSim natomiast najważniejszymi klasami są BasicSimObject oraz BasicSimEvent. Pierwsza z nich, analogicznie do DESKit, reprezentuje obiekty symulacyjne, a druga odnosi się do zdarzeń w świecie symulacyjnym. Takimi zdarzeniami są np.: rozpoczęcie lub zakończenie kroku w procesie, realizacja usługi w serwerze czy uszkodzenie serwera. Podejście zdarzeniowe to najważniejsza różnica koncepcyjna względem DESKit. Program w tym środowisku jest jednowątkowy i realizuje kod zawarty w zdarzeniach w kolejności od najwcześniejszego według zapisów zawartych w kalendarzu. Gdy czas symulacji osiągnie czas zdarzenia symulacyjnego, realizowana jest zawartość metody stateChange.

Na wysokim poziomie koncepcyjnym łatwiej jest projektować w środowisku DESKit, ponieważ bardziej odpowiada on intuicyjnemu rozumieniu świata symulacyjnego (obiekty i ich działania). Z kolei na niskim poziomie koncepcyjnym (implementacyjnym) łatwiej projektować w środowisku DISSim, ponieważ nie trzeba mieć na uwadze wielowątkowości, synchronizacji itd. Ponadto jednowątkowy program łatwiej jest debugować. Rozstrzygającym czynnikiem powodującym porzucenie środowiska DESKit na rzecz DISSim jest wydajność. W testach, w zależności od liczby instancji procesu (w DESKit – równoległych wątków), różnica w wydajności między rozwiązaniami była od kilkukrotnej (dla kilkuset procesów) do czterystukrotnej (dla 40 tys.

procesów biznesowych). Relacja ta jest zatem nieliniowa. Wynika to z potrzeby przełączania wątków, która bardziej obciąża zasoby. Ponadto przetwarzanie wielowątkowe, w tym wypadku, nie niesie za sobą korzyści, ponieważ w jednej chwili zawsze przetwarzany jest jeden wątek, a pozostałe czekają na swoją kolej. Wykorzystanie wydajnego środowiska jest bardziej wskazane, ponieważ zastosowany w optymalizatorze algorytm genetyczny wymaga oceny setek genów w setkach iteracji.

3. Model klas symulatora

Jak zaznaczono wcześniej, w środowisku DISSim najważniejszymi klasami są BasicSimObject oraz BasicSimEvent. Są to klasy abstrakcyjne, z których dziedziczyć powinny klasy biorące udział w symulacji. W przypadku symulatora systemów klasy SOA obiektami symulacyjnymi są organizacja oraz serwery. Organizacja uruchamia i zawiera w sobie instancje procesów biznesowych, zgodnie z definicją procesu. Proces biznesowy został zapisany jako graf, w którym każdy krok jest usługą. Usługi w procesie połączone są między sobą łukami opisanymi prawdopodobieństwem wyboru każdej ze ścieżek (co odzwierciedla działanie bramek w BPMN). Usługi opisane są mocą procesora i RAM-u, jakiego potrzebują do swojego działania oraz ilością danych, która powinna być przesłana przez sieć, aby usługę zrealizować. Usługi przypisane są do komponentów, które je realizują. Komponenty natomiast zawierają listę środowisk uruchomieniowych, na których mogą działać.



RYS. 1. Diagram klas symulatora systemów SOA
FIG. 1. Class diagram of SOA systems simulator

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Komponenty i środowiska są uruchamiane na serwerze, który jest obiektem symulacyjnym i opisane są mocą procesora oraz pamięcią RAM, jaka jest niezbędna do ich działania. Z kolei serwery są opisane posiadanymi zasobami niezbędnymi do uruchomienia komponentów i środowisk uruchomieniowych oraz do realizacji usług. Każdy obiekt klasy „serwer" zawiera także zmienne losowe oznaczające czas do uszkodzenia i naprawy. Ponadto serwery są opisane macierzą przepustowości sieci między nimi. Obiekty symulacyjne (serwery i organizacja) mają przypisane zdarzenia będące klasami dziedziczącymi z BasicSimEvent przedstawione na rysunku 1.

Zdarzenia uszkodzenia tworzone są po uruchomieniu symulatora w czasie wylosowanym zgodnie ze zmienną przypisaną do serwera. Gdy pojawi się zdarzenie uszkodzenia serwera, jego status zmieniany jest na niesprawny i generowane jest zdarzenie „naprawa serwera" o losowym czasie od czasu uszkodzenia. Zdarzenie naprawy generuje zdarzenie uszkodzenia itd.

4. Sposób działania

4.1. Uruchamianie procesów biznesowych

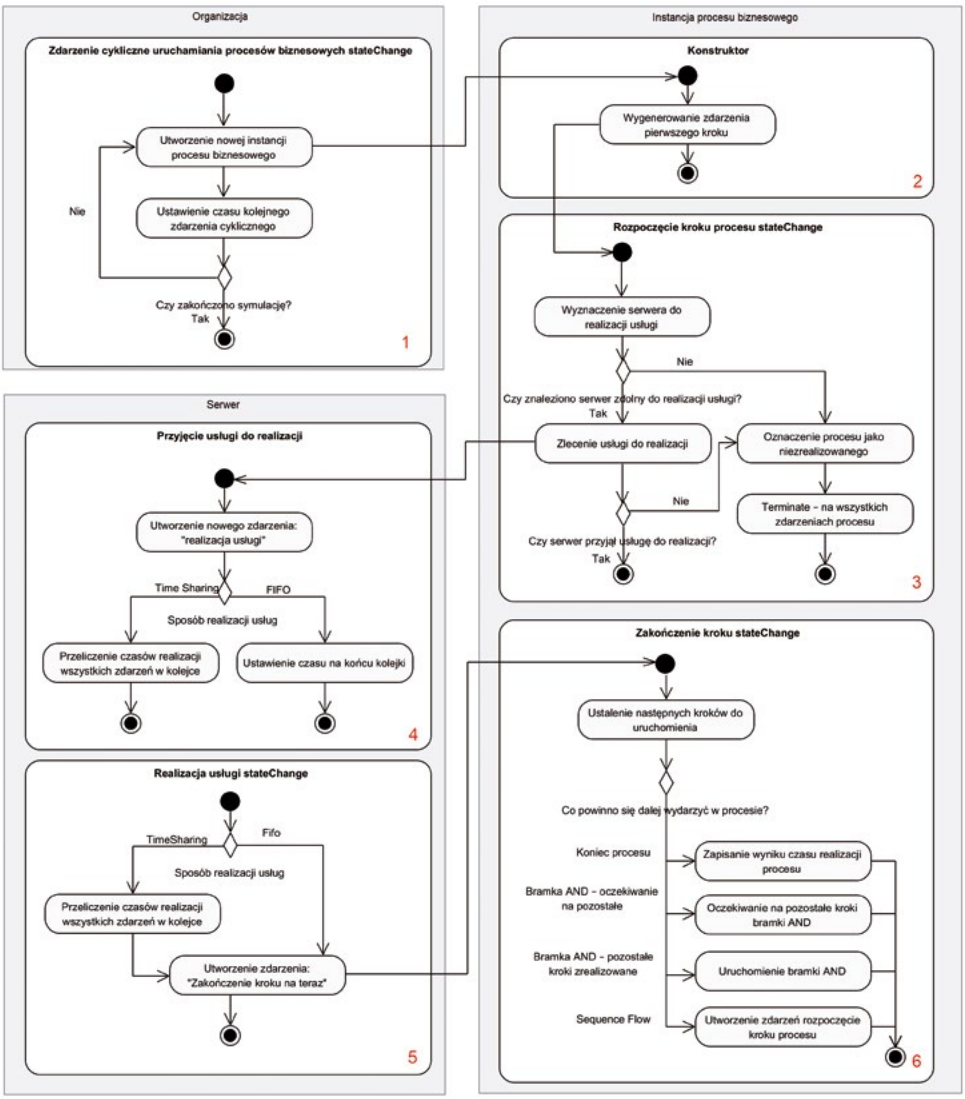
Zdarzenia w symulatorze są względem siebie zależne, a logika ich występowania została przedstawiona na rysunku 2. Po uruchomieniu symulacji generowane jest „Uruchamianie procesów biznesowych". Występuje jedno takie zdarzenie dla każdej definicji procesu biznesowego. Reprezentuje ono utworzenie nowej instancji procesu biznesowego.

Gdy tylko czas symulacji osiągnie czas zdarzenia, to tworzona jest nowa instancja procesu i nowe zdarzenie w kalendarzu, które wystąpi za losowo wygenerowany czas symulacyjny. Czas pomiędzy kolejnymi zdarzeniami uruchomienia procesu jest zdefiniowany za pomocą zmiennej losowej w definicji procesu biznesowego.

4.2. Realizacja procesu biznesowego

Gdy zostanie powołana do życia instancja procesu biznesowego, generowane jest zdarzenie realizacji jego pierwszego kroku z aktualnym czasem symulacyjnym. Każde zdarzenie reprezentujące krok w procesie biznesowym ma na celu powołanie do życia usługi, która ma ten krok zrealizować. W pierwszej kolejności wyznaczany jest właśnie ten serwer, który ma wykonać usługę. Reprezentuje to działanie load balancera, który ma wybrać właściwą instancję komponentu, czyli zrealizować algorytm Service Selection. W symulatorze zostały zaimplementowane dwie strategie Service Selection:

- najpierw serwer najmniej obciążony,
- najpierw serwer o najkrótszym spodziewanym czasie odpowiedzi.



RYS. 2. Algorytm działania symulatora
FIG. 2. The algorithm of the simulator operation

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Istnieje oczywiście możliwość rozszerzania symulatora o kolejne algorytmy Service Selection. Jeśli nie zostanie znaleziony serwer zdolny do realizacji kroku w procesie to jest on kończony, a informacja o niezdolności systemu do realizacji procesu odkładana do wyników symulacji.

4.3. Realizacja usługi

Jeśli znaleziono serwer zdolny do realizacji usługi, czyli taki, który:
- ma wystarczającą ilość wolnej pamięci RAM,
- ma uruchomiony komponent zdolny do realizacji usługi,

to tworzone jest zlecenie realizacji usługi. Czas realizacji usługi to suma czasu transferu przez sieć oraz realizacji zadania. Czas transferu jest zależny od wielkości danych do przesłania zdefiniowanych w usłudze oraz przepustowości sieci. Czas realizacji jest zależny od mocy procesora, jaką serwer może przeznaczyć na rzecz realizacji usług, liczby usług zleconych do realizacji, mocy potrzebnej do realizacji usługi oraz modelu działania komponentu (FIFO lub Time Sharing). Jeśli komponent działa w trybie Time Sharing, to każde pojawienie się nowej usługi do realizacji oraz każde zakończenie przetwarzania usługi wymaga przeliczenia oczekiwanych czasów realizacji usług.

4.4. Realizacja kroku procesu

Zakończenie realizacji usługi powoduje utworzenie zdarzenia zakończenia kroku w procesie biznesowym. Ma ono na celu ustalenie kolejnych działań w ramach procesu. Może to skutkować zakończeniem procesu lub utworzeniem zdarzeń rozpoczęcia jednego lub wielu kroków w procesie. Jeśli po kroku w procesie biznesowym użyta została jedna z bramek XOR, OR lub AND, to taki krok ma relacje do wielu kolejnych kroków w procesie. Każda relacja jest opisana prawdopodobieństwem wyboru ścieżki w przypadku bramek XOR i OR. W przypadku bramki XOR wybierany jest dokładnie jeden kolejny krok w procesie, zgodnie z prawdopodobieństwami w relacjach, które sumują się do 1. W przypadku bramki OR prawdopodobieństwo każdej bramki liczone jest niezależnie. Ich suma może być większa niż 1, a więc może zostać wybrany jeden lub wiele kroków. W przypadku bramki AND uruchamiane są zawsze wszystkie następujące kroki. Analogiczne działanie przeprowadza się w przypadku bramek łączących (występujących przed krokiem) – w tym wypadku, w celu uruchomienia kolejnego kroku, wymagane jest najpierw zakończenie się jednego lub wielu kroków poprzedzających.

5. Podsumowanie

Pomimo, że SOA jest obecnie powszechnym sposobem wytwarzania i zarządzania systemami w organizacjach, to wciąż nie została przygotowana metoda optymalizacji przydziału komponentów do serwerów, która uwzględniałaby wszystkie istotne uwarunkowania związane z tym podejściem. Została więc zaproponowana metoda oparata o algorytm genetyczny, który do oceny rozwiązania wykorzystuje symulację. Do realizacji symulatora zastosowano środowisko DISSim ze względu na jego wysoką

wydajność oraz łatwość w implementacji i debugowaniu. Przy projektowaniu symulatora opartego o zdarzenia warto zamodelować działania obiektów symulacyjnych w formie algorytmu, a następnie rozmieścić w nim zdarzenia symulacyjne.

Zaprezentowany symulator pozwala nie tylko na optymalizację przydziału komponentów do serwerów, ale także może być wykorzystywany do analizy projektowanych systemów SOA. Pozwala m.in. na:

- wykrywanie wąskich gardeł,
- ocenę ryzyka niewykonania procesu biznesowego,
- analizę wpływu wprowadzanych w systemie planowanych zmian na procesy biznesowe organizacji.

Może on zostać również wykorzystany do badań naukowych nowych algorytmów wyboru instancji usługi oraz wyboru kolejności realizacji usług w kolejkach komponentu (obecnie zaimplementowano FIFO oraz time sharing).

Literatura

1. Czarnul P. Modelling, optimization and execution of workflow applications with data distribution, service selection and budget constraints in BeesyCluster. *Computer Science and Information Technology* (*IMCSIT*); 2010.
2. Xiang C, Zhao W, Tian C, Nie J, Zhang J. *QoS-aware, Optimal and Automa-ted Service Composition with Users' Constraints*, e-Business Engineering (ICEBE), 2011, IEEE 8th International Conference.
3. Liu Y, Wu L, Liu S. *A Novel QoS-Aware Service Composition Approach Based on Path Decomposition*, Services Computing Conference (APSCC), 2012, IEEE Asia-Pacific.
4. Syu Y, FanJiang Y, Kuo J, Ma S. *Towards a Genetic Algorithm Appro-ach to Automating Workflow Composition for Web Services with Transactional and QoS-Awareness*, IEEE World Congress; 2011.
5. Ludwig S. *Clonal selection based genetic algorithm for workflow service selection,* Evolutionary Computation (CEC), 2012, IEEE Congress.
6. Oh M, Baik J, Kang S, Choi H. An Efficient Approach for QoS-Aware Servi-ce Selection Based on a Tree-Based Algorithm. *Computer and Information Science*. 2008; Seventh IEEE/ACIS International Conference.
7. Zhang C, Chang R, Perng C, So E, Tang C, Tao T. *An Optimal Capacity Planning Algorithm for Provisioning Cluster-Based Failure-Resilient Composite Services*. Services Computing, 2009. SCC IEEE International Conference.
8. Xie L, Luo J, Qiu J, Pershing J, Li Y, Chen Y. *Availability weak point analysis over an SOA deployment framework*. Network Operations and Management Symposium; 2008.

9. Almeida J, Almeida V, Ardagna D, Francalanci C, Trubian M. *Resource Management in the Autonomic Service-Oriented Architecture*, Autonomic Computing, 2006. ICAC IEEE International Conference.
10. Dyachuk D, Deters R. *Service Level Agreement Aware Workflow Scheduling*, Services Computing, 2007. SCC IEEE International Conference.
11. Dyachuk D, Deters R. *Ensuring Service Level Agreements for Service Workflows*, Services Computing, 2008. SCC IEEE International Conference.

Streszczenie

Architektura zorientowana na usługi (SOA) stała się popularna w wielu organizacjach, a pojawienie się koncepcji mikrousług wzmaga proces wdrażania jej w kolejnych firmach. W związku z tym coraz istotniejsze jest wdrażanie systemów SOA w sposób, który jak najlepiej wspiera procesy biznesowe organizacji. W opracowaniu przedstawiono symulator działania systemów SOA, który ma pozwolić na jego ewaluację. Pokrótce opisano zastosowane środowisko implementacyjne – DISSim. Następnie pokazano model klas takiego symulatora oraz główny algorytm, według którego pojawiają się kolejne zdarzenia.

Słowa kluczowe: SOA, symulacja, procesy biznesowe, optymalizacja

Summary
Simulator of SOA class systems

Service-oriented architecture (SOA) has become popular in many organizations, and the emergence of the concept of micro-services strengthens the process of its implementation in companies. Therefore, it is increasingly important to implement SOA systems in a way that best supports the organization's business processes. The paper presents a simulator of the SOA systems, which is to allow its evaluation. The implementation environment is briefly presented. Next, the class model of such a simulator and the main algorithm according to which subsequent events appear is shown.

Keywords: SOA, simulation, business processes, optimization

DOI: 10.24427/978-83-66391-28-4_11

Optymalizacja procesu magazynowania wysokoskładowego

Piotr KISIELEWSKI[*]
Przemysław TALAREK[**]

1. Wstęp

Przedsiębiorstwa biorące aktywny udział w łańcuchach dostaw dążą do poprawy swojego funkcjonowania, a przy tym do zwiększenia konkurencyjności poprzez ulepszanie funkcjonowania różnych procesów, w tym procesu magazynowania. Ulepszanie to ma prowadzić do obniżenia kosztów, zwiększenia wydajności lub ograniczenia pomyłek występujących w obsłudze magazynowej. Wprowadzanie zmian w organizacji magazynu nie może być dokonywane metodą prób i błędów, gdyż decyzje podjęte w ten sposób mogą prowadzić do występowania przestojów w pracy magazynu i w konsekwencji poważnych strat finansowych.

Gospodarka magazynowa, ze względu na złożoność procesów, jakie w niej występują, powinna być wspierana przez systemy informatyczne dopasowane do funkcji oraz charakteru docelowego magazynu, w którym mają być wykorzystywane. Przeprowadzenie symulacji procesu magazynowania na podstawie posiadanych lub przewidywanych danych wejściowych pozwala wskazać kierunek zmian w zarządzaniu magazynem, które mają na celu zoptymalizować proces magazynowania.

Do najważniejszych procesów występujących w magazynie należy kompletacja produktów, której sprawne przeprowadzenie w dużej mierze decyduje o jakości obsługi klienta. Usprawnienie tego procesu, nawet w niewielkim stopniu, może wygenerować znaczne oszczędności w przypadku magazynów obsługujących wiele zleceń dziennie. Lepszy przepływ towarów w magazynie pozwala obsłużyć więcej zamówień, co pozwala zwiększyć zyski.

W artykule przedstawiono problem optymalizacji rozkładu artykułów w magazynie wysokiego składowania, tak aby droga niezbędna do skompletowania zamówienia była jak najkrótsza. W procesie optymalizacji wykorzystano analizę materiałową ABC, algorytm Dijkstry oraz autorski program realizujący niezbędne obliczenia, napisany w języku programowania Python.

[*] Politechnika Krakowska
[**] Clonex Sp. z o.o. Sp.k.

2. Metody optymalizacji magazynowania

Magazyn, jako część łańcucha logistycznego, jest jego istotnym elementem odpowiedzialnym za przechowywanie materiałów, a następnie kierowanie ich do kolejnych ogniw tego łańcucha. W zarządzaniu magazynem istotne jest umiejętne wykorzystanie dostępnej powierzchni. Można to osiągnąć poprzez rozplanowanie wielkości i ułożenia strefy przyjęcia, magazynowania, kompletacji oraz wydawania materiału. Planowanie stref w magazynie musi uwzględniać warunki określające sposób zagospodarowania powierzchni magazynu, m.in. złożoność procesu magazynowania, zastosowane fronty przeładunkowe i ich ułożenie [1, 2, 4].

Założeniem optymalizacji magazynowania jest zwiększenie efektywności wykorzystania zasobów logistycznych magazynu. Optymalizacja może być związana ze zmianą wyposażenia magazynu, układu stref magazynowych, sposobu przydzielania zadań oraz ich wykonywania przez pracowników. Dokonując optymalizacji procesów magazynowania, należy przyjąć kryterium, według którego będzie można ocenić wpływ zmian w organizacji procesu.

Wykorzystanie powierzchni magazynowej ocenia się przez stosunek powierzchni wykorzystanej do całkowitej dostępnej powierzchni. W magazynach, gdzie nie są wykorzystywane regały paletowe, uzyskanie najwyższej wartości tego wskaźnika zapewnia składowanie materiałów w układzie blokowym, a wartości wskaźnika wykorzystania powierzchni wynoszą tutaj od 0.6 do 0.8. Dla porównania, wskaźnik ten dla rzędowego układu składowania wynosi od 0.25 do 0.6. Dążenie do optymalizacji wykorzystania powierzchni z zastosowaniem tego typu składowania powoduje ograniczenie warunków piętrzenia materiałów oraz brak dostępu do asortymentu znajdującego się w środku bloków i możliwe jest do zastosowania jedynie dla asortymentu jednorodnego, przy czym nie wymaga dodatkowych nakładów finansowych na wyposażenie magazynu. W przypadku, kiedy jedynym kryterium oceny jest zwiększenie ilości składowanego asortymentu, dobrym rozwiązaniem okazuje się wykorzystanie przepływowych regałów paletowych. Zapewniają one wysoki wskaźnik wykorzystanej powierzchni z powodu ograniczenia ilości dróg transportowych, ale jednocześnie zmuszają do wykorzystania zasady FIFO (*First In First Out* – pierwsze weszło, pierwsze wyszło). Maksymalne wykorzystanie dostępnej powierzchni magazynowej możliwe jest dzięki zastosowaniu metody wolnych miejsc składowania, która zakłada, że asortyment może być umieszczony w każdym wolnym gnieździe regałowym [1, 4].

Proces obsługi magazynowanego asortymentu pod względem czasochłonności może być ograniczony za pomocą zmiany sposobu składowania, badania częstotliwości wydania każdego produktu oraz przemyślane wyznaczanie trasy kompletacji zamówienia.

Optymalnym sposobem magazynowania jest przypisanie stałych miejsc składowania artykułu w układzie rzędowym. Materiał składowany jest niezmiennie w jednym miejscu, a grupy produktów znajdują się obok siebie w zależności od zastosowanego

kryterium podziału, takiego jak typ materiału, producent czy odbiorca. Ułożenie w ten sposób materiału umożliwia szybkie odnalezienie pożądanego produktu, co przekłada się na skrócenie czasu kompletacji zamówień [3].

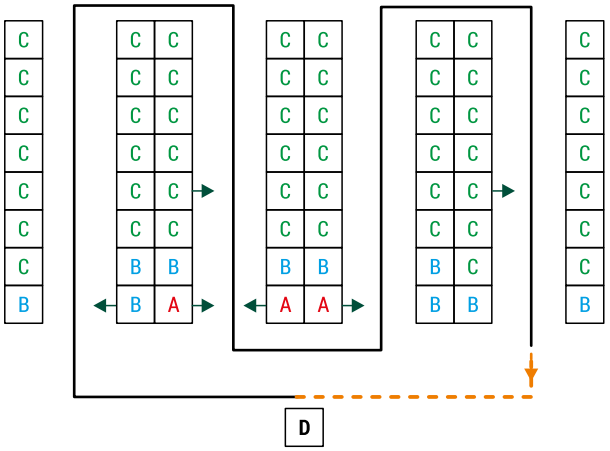
Kontrolowanie częstotliwości wydawania każdego produktu pozwala określić, które produkty powinny zostać umieszczone najbliżej doku kompletacyjnego, dzięki czemu można skrócić drogę pokonywaną w celu pobrania materiału podczas kompletacji zamówień. Głównym czynnikiem poddanym analizie jest wielkość rotacji każdego produktu, czego efekt stanowi przypisanie każdego materiału do grupy według analizy ABC, dzielącej materiały na trzy grupy:

- A – materiały najczęściej pobierane,
- B – materiały o średniej częstotliwości pobierania,
- C – materiały pobierane najrzadziej [4].

Analiza ABC opiera się o regułę Pareto 80-20, w założeniu której 80% zysków z procesu generowanych jest przez 20% asortymentu. Analizę ABC wykonuje się na podstawie zapisów dokumentacji magazynowej, według której należy obliczyć ilość wydań danego produktu, a następnie przypisać go do konkretnej grupy i określić jego lokalizację. Ta metoda wykorzystywana jest głównie w przypadku, kiedy nie występuje zmiana fizycznej postaci asortymentu w trakcie kompletacji zamówienia [3].

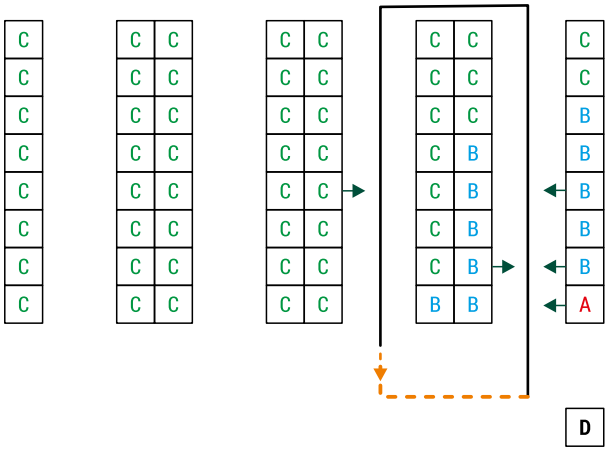
Przypisanie stałych miejsc składowania oraz wykonanie analizy ABC można poszerzyć o strategię rozlokowania asortymentu w magazynie według poniższych wytycznych, które zostały schematycznie przedstawione na rysunkach od 1 do 4:

1. Across-Aisle – artykuły umieszczane są w głąb magazynu od doku kompletacyjnego, poziom po poziomie.
2. Diagonal Storage – artykuły grupy A umieszczane są najbliżej doku kompletacyjnego, a artykuły grupy C najdalej. Do zastosowania tej strategii niezbędne jest wcześniejsze obliczenie odległości między każdym gniazdem paletowym a dokiem kompletacyjnym.
3. Perimeter Storage – artykuły powinny być umieszczane wokół magazynu, rozpoczynając od doku kompletacyjnego i idąc w głąb magazynu, a następnie ponownie umieszczane wokół magazynu.
4. Within-Aisle – strategia wykorzystywana dla magazynu, w którym dok kompletacyjny zlokalizowany jest na środku strefy kompletacyjnej. Artykuły umieszczane są po obu stronach magazynu, symetrycznie w stosunku do doku kompletacyjnego [5].



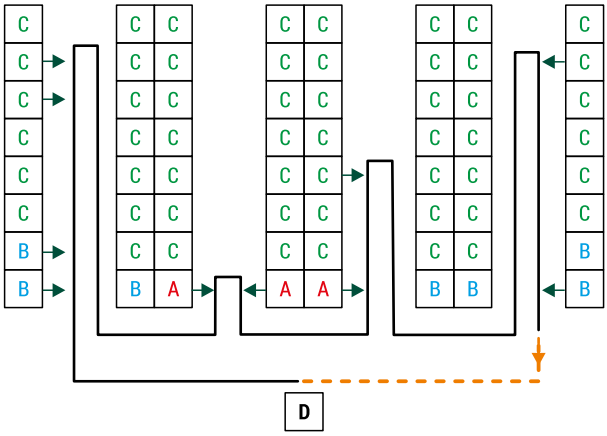
RYS. 1. Strategia Across-Aisle z metodą przejścia S-Shape
FIG. 1. Across-Aisle strategy with S-Shape routing method

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



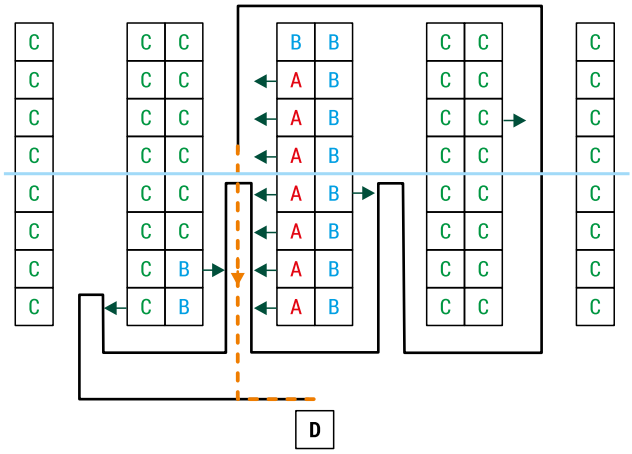
RYS. 2. Strategia Diagonal Storage z metodą przejścia S-Shape
FIG. 2. Diagonal Storage strategy with S-Shape routing method

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 3. Strategia Perimeter Storage z metodą przejścia Return
FIG. 3. Perimeter Storage strategy with Return routing method

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



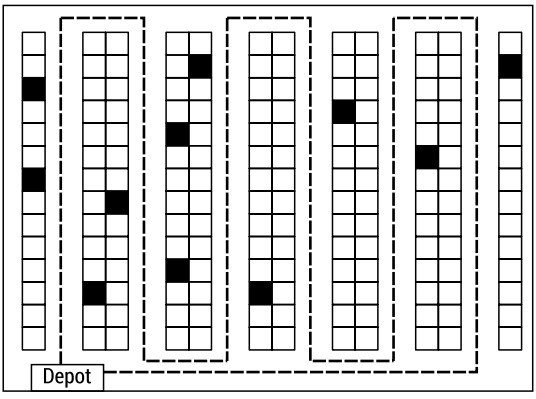
RYS. 4. Strategia Within-Aisle z metodą przejścia Midpoint
FIG. 4. Within-Aisle strategy with Midpoint routing method

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Wydajność procesu kompletacji jest w dużej mierze zależna od sposobu wyznaczania trasy dla osoby kompletującej zamówienie. Każda trasa powinna być ustalana przez system informatyczny wykorzystujący odpowiednie algorytmy deterministyczne bądź metody heurystyczne. Wykorzystanie algorytmów pozwala na znalezienie najkrótszej ścieżki, która pozwoli zrealizować dane zamówienie, natomiast wykorzystanie

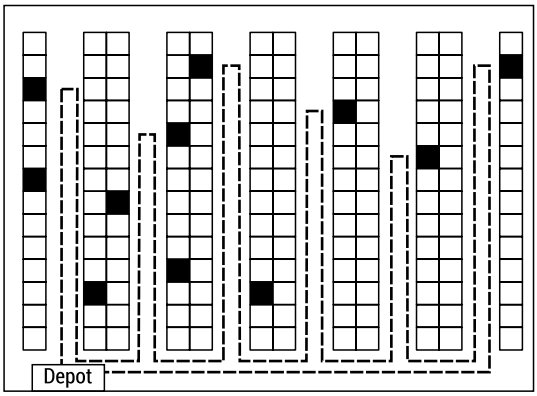
heurystyk nie zawsze gwarantuje wyznaczenie optymalnej ścieżki kompletacyjnej. Do najczęściej stosowanych tu algorytmów należy algorytm Dijkstry. Najczęściej stosowane heurystyki to:

1. Metoda S-Shape – najprostsza metoda wyznaczania trasy kompletacji. Osoba kompletująca zamówienie wchodzi do alejki z pierwszym artykułem do pobrania, a następnie przechodzi przez całą alejkę, wychodząc na drugim końcu i przechodzi do kolejnej alejki w celu pobrania artykułu. Schemat został przedstawiony na rysunku 5.
2. Metoda Return – pracownik rozpoczyna kompletację od lewej strony magazynu, pobiera wszystkie artykuły znajdujące się w alejce, a następnie cofa się do początku alejki i przechodzi do kolejnej. Schemat został przedstawiony na rysunku 6.
3. Metoda Midpoint – magazyn podzielony jest na dwie części. Pracownik dochodzi do połowy danej alejki, a następnie wraca i wchodzi do następnej. Jedynie pierwsza i ostatnia alejka pokonywane są w całości. Schemat został przedstawiony na rysunku 7.
4. Metoda Largest Gap – pracownik wchodzi do pierwszej alejki i opuszcza ją po przeciwnej stronie. W następnych alejkach osoba kompletująca zamówienie pokonuje drogę do momentu dotarcia do lokacji, z której lepszym wyjściem jest cofnięcie się do wejścia alejki, aby później wejść do niej z drugiej strony magazynu. Tylko pierwsza i ostatnia alejka w magazynie pokonywane są w całości. Schemat został przedstawiony na rysunku 8 [2].



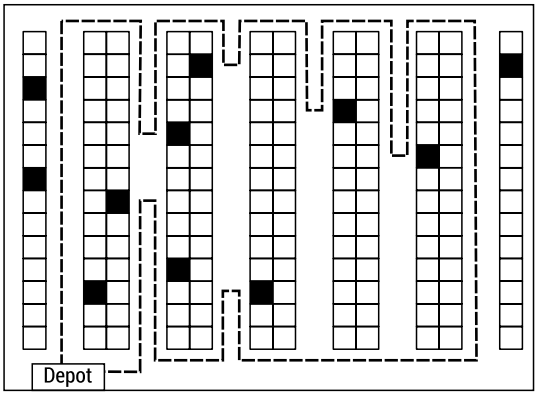
RYS. 5. Heurystyka S-Shape
FIG. 5. S-Shape heuristic

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



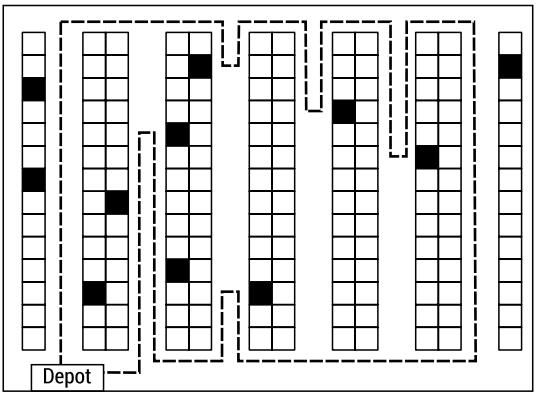
RYS. 6. Heurystyka Return
FIG. 6. Return heuristic

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 7. Heurystyka Midpoint
FIG. 7. Midpoint heuristic

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



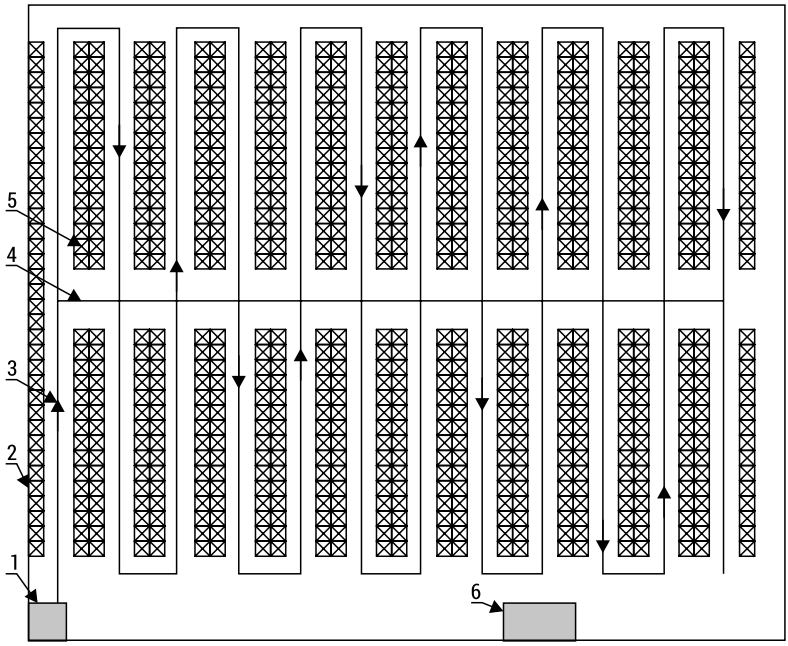
RYS. 8. Heurystyka Largest Gap
FIG. 8. Largest Gap heuristic

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

3. Przyjęty w pracy model decyzyjny

Model decyzyjny został opracowany dla magazynu wysokiego składowania należącego do firmy produkującej artykuły malarskie, takie jak farby, grunty i podkłady. Na powierzchnię magazynową składają się dwie strefy składowania produktów oddzielone od siebie drogą komunikacyjną. Każda strefa składa się z 12 alejek z 10 regałami rzędowymi po obu jej stronach. Regały stosowane w tym magazynie mają 7 m wysokości i umożliwiają składowanie materiałów na 6 poziomach po 3 palety na każdy poziom. Łączna liczba miejsc paletowych w tym magazynie wynosi 4 320, a liczba wszystkich pozycji asortymentowych jest równa 3 240 i zajmuje 3 850 miejsc paletowych, co daje 84% wykorzystania łącznej powierzchni magazynowej. Schemat magazynu został przedstawiony na rysunku 9.

Celem optymalizacji rozmieszczenia produktów w magazynie jest skrócenie drogi, jaką pokonuje osoba kompletująca zamówienia. Przyjęte zostało, że jeden pracownik w tym samym czasie może skompletować produkty należące do jednego zamówienia. Miejsce rozpoczęcia oraz zakończenia kompletacji znajdują się w dwóch różnych miejscach, a wielkość palety, na której umieszczane są pobrane materiały jest wystarczająca, aby zebrać całe zamówienie bez konieczności odłożenia pełnej palety i pobrania nowej. W ramach analizy przyjęto, że obliczenia zostaną wykonane dla zamówień składających się odpowiednio z 5, 10, 20, 30, 40 i 50 produktów. Dla każdego wariantu zostanie wygenerowanych 1 000 list kompletacyjnych.



RYS. 9. Schemat magazynu: 1 – strefa rozpoczęcia kompletacji, 2 – paleta z materiałem, 3 – droga kompletacji, 4 – droga komunikacyjna, 5 – regał przesuwny, 6 – miejsce odkładania skompletowanych zamówień

FIG. 9. Warehouse layout: 1 – order picking beginning zone, 2 – pallet with material, 3 – order picking path, 4 – communication path, 5 – sliding rack, 6 – order picking finishing zone

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Produkty będą umieszczane w regałach według strategii składowania Diagonal Storage. Każdy produkt w magazynie ma przypisaną współrzędną *x, y, z*, które informują o jego lokalizacji w konkretnej alejce, na odpowiednim regale oraz wysokości składowania. Współrzędne służą do obliczenia odległości między kolejnymi punktami, do których musi udać się pracownik magazynu.

Współrzędna *X*:

1. określa alejkę, w której znajduje się pożądany produkt;
2. pierwsza alejka ma przypisaną współrzędną $x = 2$, ponieważ $x = 1$ odnosi się do lokalizacji pierwszego regału z którego pobierany jest produkt;
3. każda następna alejka ma współrzędną x zwiększoną o 3 w stosunku do poprzedniej, czyli $x_i = x_{i-1} + 3$ dla $i > 1$;
4. w rozpatrywanym magazynie jest 12 alejek, więc $X = [2, 5, \ldots, 35]$.

Współrzędna *Y*:

1. określa długość strefy kompletacyjnej;
2. miejsce składowania pierwszej palety w alejce zaczyna się od $y = 11$, $y = [1, 2, \ldots, 10]$ i odnosi się do lokalizacji „wejścia" – wózki widłowe służące do kompletacji zamówień wyposażone są w czujnik magnetyczny, który nie pozwoli na wjazd do alejki bez odpowiedniego ustawienia wózka widłowego względem ścieżki magnetycznej wzdłuż regałów;
3. miejsce składowania każdej następnej palety wzdłuż alejki oznaczone jest jako $y_i = y_{i-1} + 1$;
4. w rozpatrywanym magazynie jest 30 regałów wysokiego składowania na każdą stronę alejki;
5. współrzędne do wykorzystania przez algorytm przy rozmieszczaniu produktów mieszczą się w zakresie $y = [11, 12, \ldots, 45]$ oraz $y = [49, 50, \ldots, 83]$;
6. współrzędne *Y* w zakresie $y = [46, 47, 48]$ zarezerwowane są dla drogi komunikacyjnej między strefami kompletacyjnymi.

Współrzędna *Z*:

1. określa ilość poziomów składowania;
2. w rozpatrywanym magazynie jest sześć poziomów składowania;
3. miejsce składowania pierwszej palety w alejce zaczyna się od $z = 1$; współrzędne do wykorzystania przez algorytm przy rozmieszczaniu produktów mieszczą się w zakresie $z = [1, 2, \ldots, 6]$.

W modelowaniu magazynu, w celu usprawnienia oraz ułatwienia obliczeń, przyjęto, że droga pokonywana przez magazyniera zostanie podzielona przez drogę odniesienia równą odległości między środkiem geometrycznym dwóch sąsiadujących gniazd paletowych – $L_{odn} = 1{,}4$ [m]. W wyniku tego zabiegu miary odległościowe zostały w projekcie sprowadzone do wielkości bezwymiarowych.

Aby określić, które produkty należy umieścić jak najbliżej doku kompletacyjnego, posłużono się metodą zarządzania zapasami, a dokładnie analizą ABC. Kryterium, według którego dokonano tej analizy, jest częstość występowania danego produktu na zamówieniu kompletacyjnym. Sposób wyznaczania tej wielkości przedstawia wzór (1).

$$\delta_m = \frac{I_m}{I} \tag{1}$$

gdzie:

δ_m – częstość występowania produktu *m* na listach kompletacyjnych w badanym okresie;
I_m – liczba list kompletacyjnych, na których występuje produkt *m* w badanym okresie;
I – liczba wszystkich list kompletacyjnych w badanym okresie.

Oceny proponowanej zmiany w rozmieszczeniu produktów dokonujemy według kryterium opisanego wzorem (2). Sposób obliczenia średniej odległości, jaką musi przebyć magazynier podczas kompletacji zamówienia, został opisany wzorami (3) oraz (4).

$$\Delta = \left(1 - \frac{L_2}{L_1}\right) 100\% \tag{2}$$

gdzie:

Δ – kryterium oceny wprowadzonych zmian w rozmieszczeniu produktów w magazynie, [%];

L_1 – średnia odległość, jaką musi przebyć osoba kompletująca zamówienie przed zmianą w rozmieszczeniu produktów, określona wzorem:

$$L_1 = \frac{1}{I} \sum_{n=1}^{I} l_n \tag{3}$$

L_2 – średnia odległość, jaką musi przebyć osoba kompletująca zamówienie po zmianie w rozmieszczeniu produktów, określona wzorem:

$$L_2 = \frac{1}{I} \sum_{n=1}^{I} l_n^p \tag{4}$$

W powyższych wzorach przyjęto oznaczenia:

l_n – odległość, jaką pokonuje osoba kompletująca *n*-te zamówienie przed zmianą rozmieszczenia produktów;

l_n^p – odległość, jaką pokonuje osoba kompletująca *n*-te zamówienie po zmianie rozmieszczenia produktów;

I – liczba wszystkich list kompletacyjnych w badanym okresie.

W celu obliczenia drogi, jaką pokonuje osoba kompletująca zamówienie, przyjęto, że musi ona odwiedzić przynajmniej pięć miejsc w magazynie:

1. miejsce poboru pustej palety;
2. lokalizację „wejścia" do alejki, w której znajduje się pożądany produkt;
3. miejsce składowania produktu;
4. lokalizację „wyjścia" z alejki, w której obecnie znajduje się pracownik;
5. miejsce odłożenia palety z zamówieniem.

W przypadku wystąpienia na liście kompletacyjnej produktów, które znajdują się w różnych alejkach, po punkcie 4 należy wykonać ponownie kroki 2-4. Wszystkie odwiedzane miejsca zapisywane są w jednej tabeli, co umożliwia obliczenie odległości między każdą parą sąsiadujących ze sobą lokalizacji.

Do obliczeń wykorzystano wzór (5):

$$l_i = \sqrt[2]{\left(x_{i+1} - x_i\right)^2 + \left(y_{i+1} - y_i\right)^2 + \left(z_{i+1} - z_i\right)^2}, \quad i = j - 1 \tag{5}$$

gdzie:
l_i – odległość między dwoma punktami na liście kompletacyjnej;
x_i – współrzędna x dla pierwszej lokalizacji, dla której obliczana jest odległość;
y_i – współrzędna y dla pierwszej lokalizacji;
z_i – współrzędna z dla pierwszej lokalizacji;
x_{i+1} – współrzędna x dla drugiej lokalizacji, dla której obliczana jest odległość;
y_{i+1} – współrzędna y dla drugiej lokalizacji;
z_{i+1} – współrzędna z dla drugiej lokalizacji;
j – liczba miejsc, do których musi udać się magazynier przy kompletacji zamówienia.

Całkowitą odległość, jaką pokonuje osoba kompletująca zamówienia przedstawia wzór (6):

$$l_n = \sum_{i=1}^{j-1} l_i \tag{6}$$

l_n – łączna odległość do przebycia przy kompletacji n-tego zamówienia.

4. Badania symulacyjne

Algorytm wykorzystany w aplikacji pozwala na obliczenie drogi, jaką pokonuje osoba kompletująca zamówienie w magazynie o dowolnych wymiarach, jednak przy zachowaniu pewnych stałych elementów, takich jak:
1. miejsce rozpoczęcia kompletacji;
2. miejsce zakończenia kompletacji;
3. podział magazynu na dwie strefy oraz istnienie drogi komunikacyjnej między tymi strefami;
4. miejsce, w którym położony jest pierwszy regał oraz kierunku poszerzania strefy magazynowania.

Program symulacyjny miał za zadanie:
1. określić parametry magazynu;
2. określić pulę miejsc składowania produktów w magazynie;
3. określić asortyment dostępny w magazynie;
4. utworzyć listy kompletacyjne;
5. opisać asortyment w magazynie za pomocą analizy ABC;
6. obliczyć odległość, jaką pokonuje magazynier przy kompletowaniu zamówień, wykorzystując każdą metodę wyznaczania trasy i zapisać wyniki;
7. dokonać optymalizacji ułożenia asortymentu w magazynie według analizy ABC.

Do wyznaczenia ścieżki, po jakiej porusza się osoba kompletująca zamówienia, wykorzystano algorytm Dijkstry, heurystykę S-Shape, Midpoint, Return, a także Buyer i Sorted-Buyer. Dwa ostatnie sposoby opisują zbieranie zamówień w sposób podobny do robienia zakupów w hipermarkecie.

Metoda Buyer przedstawia pobieranie artykułów przez osobę, która nie zna rozłożenia produktów w sklepie i produkty na jej liście są ułożone w sposób losowy. Metoda Sorted-Buyer przedstawia zachowanie osoby, która doskonale orientuje się w ułożeniu asortymentu w sklepie, przez co jej lista zakupów jest ułożona w taki sposób, aby nie musiała chodzić kilka razy przez daną alejkę z produktami, aby pobrać interesujący ją artykuł.

Wyznaczanie trasy do pokonania z użyciem algorytmu Dijkstry odbywa się w następujących krokach:

1. Utwórz macierz najmniejszych odległości między każdym punktem na liście kompletacyjnej, włącznie z miejscem rozpoczęcia i zakończenia kompletacji.
2. Pogrupuj lokacje z listy kompletacyjnej według alejki, której dotyczą oraz strefy magazynu, tworząc sektory.
3. Utwórz listę decyzyjną zawierającą po jednej cyfrze przypadającej na każdy sektor, którą następnie wprowadź do zbioru list decyzyjnych.
4. Oblicz odległość, jaką trzeba pokonać z pierwszego do drugiego sektora, pobierając przy tym wszystkie produkty z drugiego sektora, uwzględniając dwa warianty kolejności, w jakiej pobiera się artykuły i wybierz opcję pozwalającą na skrócenie wymaganego dystansu:
   a) gdy elementy sektora posortowane są według współrzędnej *y* w rosnącej kolejności,
   b) gdy elementy sektora posortowane są według współrzędnej *y* w malejącej kolejności.
5. Powtórz krok 4 dla każdego następnego sektora, zapisz otrzymane wyniki.
6. Zamień w sposób losowy kolejność sektorów, do których musi udać się pracownik magazynu (poza pierwszym i ostatnim), a następnie nowo powstałą listę decyzyjną wprowadź do zbioru list decyzyjnych (jeżeli nowa lista obecnie znajduje się w zbiorze, powtórz krok 6).
7. Powtórz krok 4 oraz 5, a następnie wybierz wariant dający lepszy wynik.
8. Powtarzaj kroki 4-7 aż do osiągnięcia 10 000 iteracji.

5. Wyniki

Dzięki wykorzystaniu modelu decyzyjnego opisanego w rozdziale 3 oraz algorytmu w rozdziale 4, otrzymano wyniki przedstawiające możliwy do osiągnięcia poziom optymalizacji magazynowania wysokoskładowego. Wyniki zostały przedstawione w sposób zbiorczy w tabeli 1. Różnica w odległości, jaka jest niezbędna do pokonania przy kompletacji zamówień przed i po alokacji produktów metodą analizy ABC,

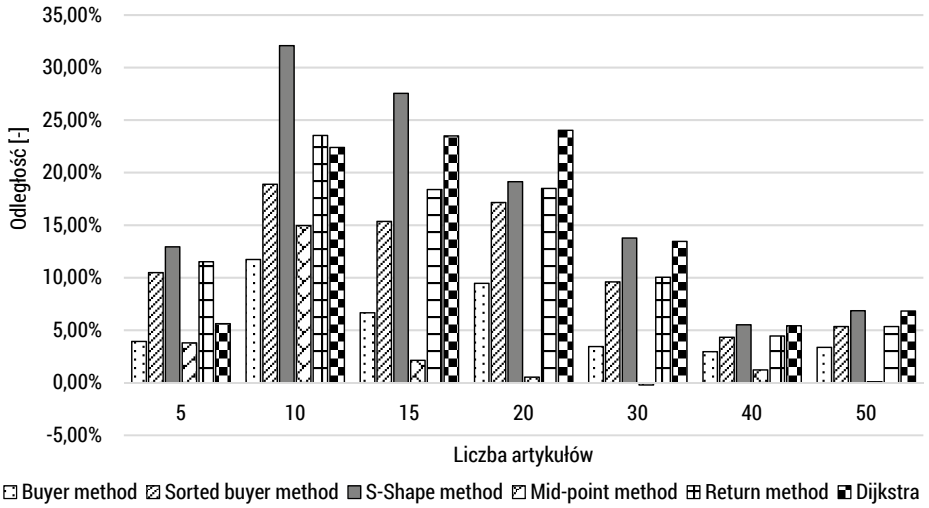
została przedstawiona w sposób jednostkowy i procentowy. Poziom dokonanej optymalizacji kompletacji zamówień prze wykorzystaniu różnych metod wyznaczania drogi kompletacji zamówienia i dla różnej wielkości zamówień został przedstawiony na rysunku 10.

TAB. 1. Średnia odległość niezbędna do kompletacji zamówienia – wyniki zbiorcze
TAB. 1. Average distance needed for order picking – summary list

Heurystyka		Liczba artykułów:						
		50	40	30	20	15	10	5
L_1 [–]	Metoda Buyer	755,096	611,284	463,019	326,555	248,738	176,93	102,683
	Metoda Sorted buyer	185,373	176,857	165,254	150,153	136,222	123,456	89,875
	Metoda S-Shape	144,979	137,814	129,98	121,377	115,468	111,158	94,755
	Metoda Mid-point	189,037	180,601	170,264	154,842	146,078	134,216	99,896
	Metoda Largest gap	185,412	177,724	167,582	154,088	143,971	137,473	107,483
	Dijkstra	145,259	138,239	129,228	119,945	110,261	99,453	76,092
L_2 [–]	Metoda Buyer	729,597	593,212	447,045	295,647	232,151	156,165	98,645
	Metoda Sorted buyer	175,437	169,184	149,373	124,379	115,297	100,137	80,451
	Metoda S-Shape	135,039	130,207	112,075	89,865	83,654	75,482	82,500
	Metoda Mid-point	188,858	178,382	170,629	154,012	142,945	114,139	96,098
	Metoda Largest gap	175,489	169,805	150,74	125,58	117,486	105,105	95,091
	Dijkstra	135,336	130,732	111,84	91,128	84,356	77,17	71,813

Heurystyka		Liczba artykułów:						
		50	40	30	20	15	10	5
$L_2 - L_1$ [–]	Metoda Buyer	–25,499	–18,072	–15,974	–30,908	–16,588	–20,765	–4,038
	Metoda Sorted buyer	–9,936	–7,674	–15,881	–25,774	–20,926	–23,319	–9,424
	Metoda S-Shape	–9,941	–7,606	–17,905	–31,513	–31,813	–35,675	–12,255
	Metoda Mid-point	–0,179	–2,22	0,365	–0,83	–3,133	–20,077	–3,798
	Metoda Largest gap	–9,923	–7,919	–16,842	–28,509	–26,486	–32,368	–12,392
	Dijkstra	–9,923	–7,507	–17,388	–28,816	–25,905	–22,283	–4,28
Δ [%]	Metoda Buyer	3,38	2,96	3,45	9,46	6,67	11,74	3,93
	Metoda Sorted buyer	5,36	4,34	9,61	17,17	15,36	18,89	10,49
	Metoda S-Shape	6,86	5,52	13,78	25,96	27,55	32,09	12,93
	Metoda Mid-point	0,09	1,23	–0,21	0,54	2,14	14,96	3,80
	Metoda Largest gap	5,35	4,46	10,05	18,50	18,44	23,54	11,53
	Dijkstra	6,83	5,43	13,46	24,03	23,49	22,41	5,62

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.



RYS. 10. Poziom dokonanej optymalizacji dla różnych metod kompletacji zamawiania
FIG. 10. Optimization level for different methods of order completion

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Wpływ sortowania listy kompletacyjnej według artykułów do pobrania został liczbowo przedstawiony w tabeli 2. Wielkość ΔL jest różnicą między wynikami dla metod Buyer i Sorted Buyer, natomiast ΔL [%] wyraża, o ile procent wynik został poprawiony dzięki zastosowaniu sortowania.

Najlepsze wyniki w analizowanym magazynie wysokiego składowania osiągnieto przy zastosowaniu heurystyki S-shape oraz algorytmu Dijkstry. Metoda S-shape bardzo dobrze sprawdziła się, gdy trzeba było skompletować dużą ilość asortymentu lub artkuły znajdowały się w sąsiednich alejkach. Wykorzystanie analizy ABC ograniczyło o 32% niezbędną drogę do pokonania dla tej metody.

Algorytm Dijkstry okazał się lepszy w przypadku, gdy artykuły do skompletowania znajdowały się daleko od siebie lub gdy było ich mało do pobrania. W tym wypadku udało się ograniczyć drogę kompletacji o 24%.

Metoda Buyer, która opisuje pobieranie artykułów według kolejności odnotowanego zapotrzebowania na dany produkt i nie uwzględnia w żadnym stopniu optymalnego ich kompletowania, jedynie w przypadku wystąpienia pięciu artykułów na liście kompletacyjnej nie odbiegała efektywnością od pozostałych heurystyk. Znaczna różnica w efektywności zaczyna występować, gdy na liście kompletacyjnej do pobrania jest co najmniej 10 artykułów, również po dokonaniu korekty ABC ułożenia materiałów w magazynie.

TAB. 2. Wpływ sortowania artykułów na długość drogi kompletacyjnej
TAB. 2. Impact of article sorting on distance of order picking path

Przed zastosowaniem analizy ABC							
Heurystyka	Liczba artykułów:						
	5	10	15	20	30	40	50
Metoda Buyer	102,68	176,93	248,74	326,56	463,02	611,28	755,10
Metoda Sorted buyer	89,875	123,46	136,22	150,15	165,25	176,86	185,37
ΔL [–]	12,808	53,474	112,52	176,40	297,77	434,43	569,72
ΔL [%]	12,47	30,22	45,23	54,02	64,31	71,07	75,45
Po zastosowaniu analizy ABC							
Heurystyka	Liczba artykułów:						
	5	10	15	20	30	40	50
Metoda Buyer	98,645	156,17	232,15	295,65	447,05	593,21	729,60
Metoda Sorted buyer	80,451	100,14	115,30	124,38	149,37	169,18	175,44
ΔL [–]	18,19	56,03	116,85	171,27	297,67	424,03	554,16
ΔL [%]	18,44	35,88	50,34	57,93	66,59	71,48	75,95

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Dla analizowanego magazynu oraz utworzonych list kompletacyjnych heurystyka Mid-point uzyskała znacząco gorsze wyniki w obu przypadkach ułożenia asortymentu. Wpływ na efektywność tej heurystyki ma przyjęta strategia układania materiału wewnątrz magazynu. Po zastosowaniu analizy ABC wyniki otrzymane dla 30 artykułów świadczą o pogorszeniu efektywności kompletacji zamówień, a dla 50 artykułów – o znikomym wpływie na kompletację. Z tego powodu, metoda Mid-point okazała się niewskazana do stosowania w połączeniu ze składowaniem typu Diagonal Storage.

Przed analizą ABC ułożenia asortymentu w magazynie heurystyka Return pozwoliła na uzyskanie zbliżonych rezultatów co Mid-point method. Po zastosowaniu analizy ABC udało się ograniczyć nakład pracy magazyniera o ponad 20%.

6. Wnioski

Magazynowanie jest niezwykle istotnym procesem dotyczącym łańcucha dostaw. Dążenie do poprawy jego funkcjonowania należy do kluczowych zadań kadry logistycznej zarządzającej łańcuchem dostaw. Osiągniecie tego możliwe jest m.in. poprzez analizę zapotrzebowania na każdy asortyment w magazynie oraz porównanie możliwych do zastosowania metod wyznaczania trasy kompletacyjnej i sposobu rozmieszczenia materiału wewnątrz magazynu.

Przeprowadzona optymalizacja magazynowania wysokoskładowego wykazała, że kompletacja zamówień dzięki analizie materiałowej ABC może zostać polepszona nawet o 32% poprzez odpowiednie ułożenie asortymentu w magazynie.

Dokonanie rozmieszczenia materiałów według analizy ABC tylko dla jednej metody wyznaczania trasy nie przyniosło poprawy kompletacji. Wyznaczając trasę do skompletowania 30 artykułów, dla heurystyki Mid-point uzyskano o 0,21% gorsze wyniki. Również w przypadku kompletowania 50 artykułów wpływ relokacji materiałów na efektywność tej metody był bardzo mały – udało się skrócić drogę magazyniera tylko o około 0,09%, tj. w granicach błędu obliczeń.

W świetle wyników przeprowadzonych badań wykazano, że nie można jednoznacznie wytypować najlepszej metody wyznaczania drogi kompletacyjnej dla każdego zamówienia w analizowanym magazynie. Wybranie najlepszej drogi kompletacyjnej powinno zacząć się od określenia ilości artykułów do skompletowania, a następnie wybrania metody S-Shape lub algorytmu Dijkstry.

Otrzymane wyniki mają ograniczone zastosowanie dla innego typu składowania w magazynie. Różnice wyników mogą być spowodowane zbyt małą liczbą zamówień kompletacyjnych analizowanych podczas pracy algorytmu, rozmiarem magazynu i ułożeniem regałów w magazynie, sposobem losowania artykułów podczas tworzenia list kompletacyjnych, a także zastosowaną strategią rozmieszczenia materiału. W celu sprawdzenia wyników, przy uwzględnieniu tych samych list kompletacyjnych, można dokonać kolejnych symulacji, zmieniając po jednym czynniku i porównać otrzymane wyniki z bazowymi, co pozwoli dodatkowo określić wpływ wyżej wymienionych elementów na efektywność kompletacji zamówień.

Literatura

1. Niemczyk A. *Zapasy i magazynowanie. Tom II Magazynowanie. Podręcznik do kształcenia w zawodzie technik logistyk*. Biblioteka Logistyka. Poznań; 2008.
2. Roodbergen KJ. *Layout and routing methods for warehouses*. ERIM Ph.D. series Research in Management 4, Erasmus University Rotterdam, Rotterdam; 2001.
3. Kudelska I. *Metoda wyboru zmiennych miejsc składowania w magazynie*, rozprawa doktorska. Poznań; 2016

4. Dudziński Z, Kizyn M. *Vademecum gospodarki magazynowej*. Gdańsk: Ośrodek Doradztwa i Doskonalenia Kadr Sp. z o.o.; 2002.
5. Garbacz M, Łopuszański M. *Optymalizacja procesu kompletacji w magazynie* (*cz. 1*). Logistyka. 2015; 6: 628-636.

Streszczenie

W opracowaniu przedstawiono sposoby optymalizacji procesu magazynowania wysokiego składowania pod kątem skrócenia czasu kompletacji zamówień. Celem pracy jest optymalizacja procesu przechowywania wysokiego poziomu na wybranym przykładzie z użyciem oryginalnego programu napisanego w języku programowania Python. W tym celu program wykorzystuje algorytm Dijkstry, analizę materiałową ABC i przyjęty model matematyczny.

Praca składa się z dwóch części. Pierwsza koncentruje się na podstawowej teorii magazynowania, optymalizacji i realizacji zamówień. W tej części wyjaśniono również główny aspekt analizy materiałowej, który wykorzystuje regułę Pareto do optymalizacji magazynowania i algorytm Dijkstry do określania ścieżki realizacji zamówienia.

Druga część pracy zawiera kluczowe informacje, które są niezbędne do przeprowadzenia procesu optymalizacji. Na początku przedstawiono obszar magazynowy wybranego przykładu z danymi o korytarzach, strefach składowania, regałach paletowych, ilości artykułów i wykorzystaniu gniazd paletowych. Jako uzupełnienie tej informacji przedstawiono schemat techniczny w celu zilustrowania analizowanego przykładu. Następnie sformułowano założenia optymalizacyjne, określające układ magazynu, jego wymiary i wyposażenie. Ta część pracy obejmuje również wyniki symulacji komputerowej, które przedstawiono w tabelach i wykresach. W oparciu o wyniki symulacji można stwierdzić, że aspekt metody wyznaczania najlepszej drogi kompletacji zamówienia jest bardzo istotny, jeśli kierownictwo próbuje obniżyć koszty w firmie. Zastosowanie najprostszej metody Buyer do wyznaczania drogi kompletacji może skutkować skróceniem drogi nawet o 76% dzięki organizacji kolejności produktów na liście zamówienia. Wynik drogi kompletacji z najbardziej popularnych metod S-Shape można polepszyć o 32% przy użyciu analizy materiałowej ABC i o 24% przy użyciu algorytmu Dijkstry. Symulacje dla analizowanego typu magazynu ze składowaniem produktów typu Diagonal Storage wykazały problemy tylko z metodą Mid-Point, w której uzyskano gorsze wyniki po zmianie ABC alokacji produktów, ale przyczyną takich wyników mogła być niewystarczająco liczna lista produktów do kompletacji zamówienia lub strategia składowania zastosowana w przedmiotowym magazynie.

W pracy krótko scharakteryzowano analizę materiałową ABC, sposoby wyznaczania tras kompletacyjnych oraz układania asortymentu wewnątrz magazynu. Na wybranym przykładzie zaprezentowano model decyzyjny oraz algorytm, którego zadaniem było ograniczenie długości drogi niezbędnej do kompletacji zamówienia. Wyniki badań symulacyjnych zestawiono porównawczo w tabelach i przedstawiono zbiorczo na wykresach. Uzyskane wyniki umożliwiły sformułowanie podsumowujących wniosków.

Słowa kluczowe: magazynowanie wysokoskładowe, kompletacja zamówień, optymalizacja procesu

Summary
Optimization of the high-level storage process

The purpose of the paper is the optimization of the high-level storage process on a selected example with usage of original program written in Python programming language. The program uses Dijkstra's algorithm, material analysis of ABC and adopted mathematical model in order to achieve the target.

The work consists of two parts. First one is focusing on the basic theory of warehousing, optimization and completion of orders. This paragraph also explains the main aspect of material analysis which uses Pareto's rule for the storage optimization and Dijkstra's algorithm for determining the path of order completion.

Second part of the paper contains crucial information what is necessary to conduct the optimization process. At the beginning, warehousing area of selected example was presented with data about aisles, storage zones, pallet rackings, articles amount and pallet nests usage. As a supplement to this information, technical drawing was presented for the purpose of illustrating of the analyzed example. Subsequently the optimization assumptions were formulated such as dimensionless distance quantities, completion properties and restrictions, designation of warehouse layout. This part shows the results of simulation. The simulation results was presented in tables and charts. Based on the calculations, aspect of routing method is very crucial if management tries to reduce cost in a company. The most simple routing Buyer method, can be improved even by 76% just by organizing the order of products on the lists. One of the most common routing S-shape method, can be optimized by 32% with usage of material analysis of ABC and 24% with usage of Dijkstra's algorithm. There were problems only with Mid-point method, which scores says that it gives worse results after products relocation, but that might happen because amount of order picking lists were not enough or used storage strategy.

Keywords: high-level warehousing, completion of orders, optimization of process

DOI: 10.24427/978-83-66391-28-4_12

Symulacyjne badanie efektywności funkcjonowania przedsiębiorstwa serwisującego odzież roboczą

Tadeusz NOWICKI[*]
Robert WASZKOWSKI[*]

1. Wprowadzenie

W czasach dynamicznego rozwoju teorii, metod i środowisk programowych związanych z obsługą i zarządzaniem procesami biznesowymi [1, 2, 3, 4] daje się zauważyć znaczący wzrost świadomości procesowej i zwiększenie dojrzałości modeli działania wielu przedsiębiorstw na świecie, w tym również w Polsce. Skutkuje to tym, że istotnie wzrasta świadomość przedsiębiorców w zakresie konieczności tworzenia dojrzałych modeli dla realizacji procesów biznesowych i dalszej ich pielęgnacji, to znaczy takich procesów, które opisują w pełni funkcjonowanie przedsiębiorstwa. Podejście procesowe przestało być domeną dużych firm i instytucji, a zaczęło być stosowane również w średniej wielkości przedsiębiorstwach.

W pracy zaprezentowano modele procesów biznesowych przedsiębiorstwa produkującego, wynajmującego i utrzymującego w stałym obrocie kilkaset tysięcy sztuk odzieży roboczej, w tym oferujące usługę prania odzieży i jej renowacji. Modele te opracowane zostały w celu usprawnienia efektywności działania przedsiębiorstwa.

Skonstruowane i zweryfikowane procesy biznesowe zostały poddane symulacyjnemu badaniu w celu poznania wartości parametrów dynamicznych ich realizacji. Pozyskane z eksperymentów symulacyjnych wyniki wspomagają analizę prowadzącą do usprawnienia pracy przedsiębiorstwa i, co za tym idzie, zmniejszenia kosztów operacyjnych jego działalności. Jako ilustracja możliwości opisanej metody użyte zostały wybrane procesy biznesowe automatyzujące pracę działu obsługi klienta.

Usługa rentalu, czyli wynajmu i serwisu odzieży roboczej i ochronnej, to coraz popularniejsze rozwiązanie rynkowe (propozycja od wyspecjalizowanych w tym zakresie firm), niwelujące konieczność inwestowania przez przedsiębiorstwa w jednorazowy zakup i zasoby zapewniające utrzymanie odzieży roboczej. Dzięki takiemu podejściu akceptowane są miesięczne koszty związane z zaopatrzeniem pracowników

[*] Wojskowa Akademia Techniczna

w odzież pracowniczą i roboczą oraz inne potrzebne materiały tekstylne. Najczęściej równoważą się one z pomijanymi w takim przypadku kosztami utrzymywania własnych służb serwisujących odzież roboczą.

Niezależnie od odbiorcy usługi rentalu odzieży roboczej sposób jej realizacji jest podobny i zawiera w sobie usługę prania, dostarczania czystej i odbioru brudnej odzieży w określonych dniach. Dodatkowo czynione są także usługi jej napraw lub wymiany.

Przedstawione metody i wyniki projektowania procesów biznesowych i ich badań stanowią element realizacji zadań w ramach projektu „Opracowanie inteligentnego systemu zarządzania procesami biznesowymi z uwzględnieniem rozwiązań inteligentnych systemów wspierających integrację tradycyjnych oraz elektronicznych kanałów sprzedaży w rentalu odzieży roboczej (sektor B2B)" współfinansowanego z Europejskiego Funduszu Rozwoju Regionalnego w ramach Osi Priorytetowej I „Wykorzystanie działalności badawczo-rozwojowej w gospodarce" Działania 1.2 „Działalność badawczo-rozwojowa przedsiębiorstw" regionalnego programu operacyjnego województwa mazowieckiego na lata 2014-2020.

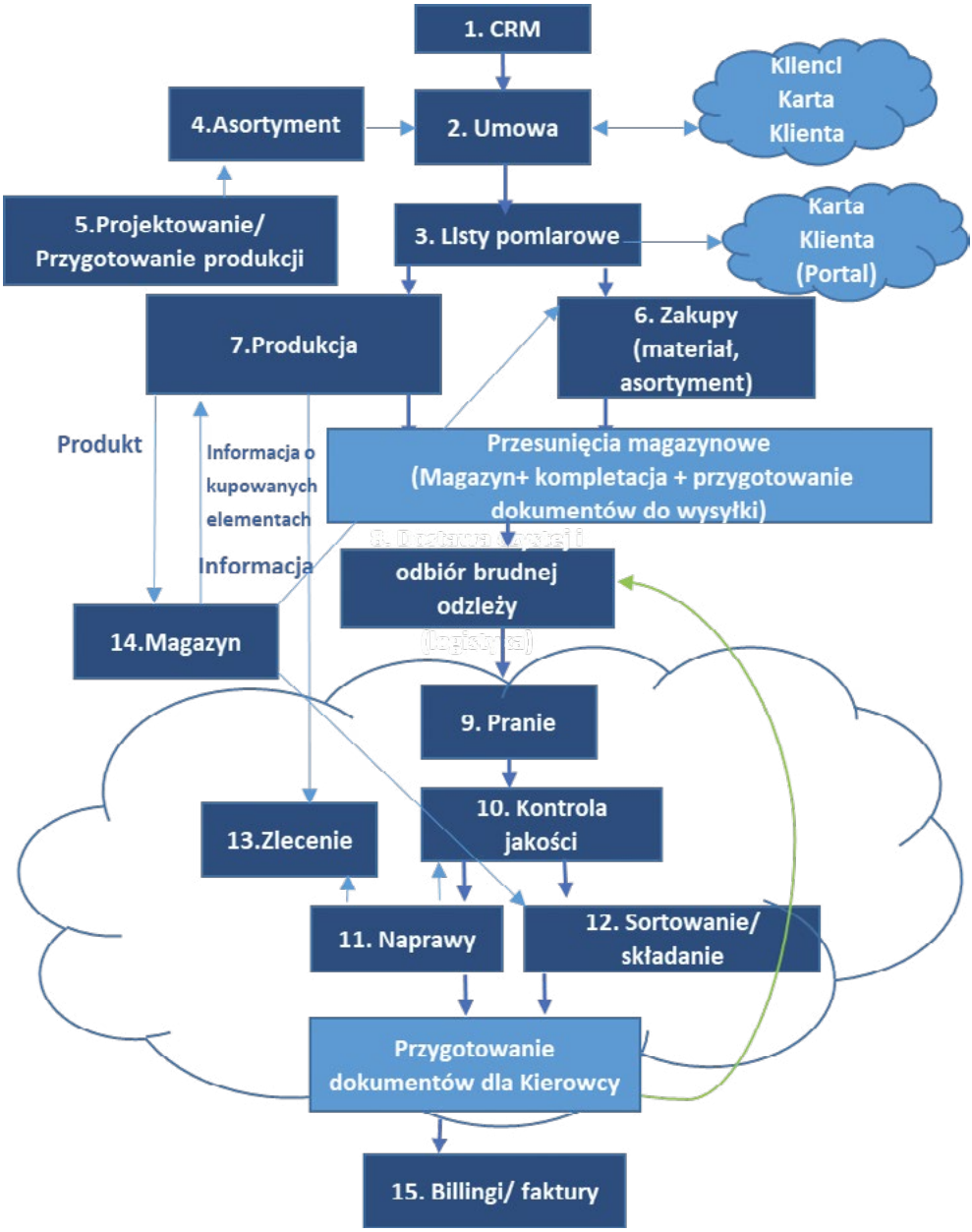
2. Procesy biznesowe przedsiębiorstwa serwisującego odzież roboczą

Opracowane w ramach projektu procesy biznesowe stanowią formalny opis działań przedsiębiorstwa obsługującego wynajem, pranie i naprawy odzieży roboczej oraz są wstępem do wytworzenia oprogramowania wspomagającego te działania. Biorąc pod uwagę to, że zamiarem beneficjenta było pozyskanie środowiska, w którym możliwe jest doskonalenie procesów i działań przedsiębiorstwa, to w konstrukcji takiego rozwiązania opracowane procesy biznesowe pełnią podstawową rolę.

Na rysunku 1 przedstawiono zakres działań przedsiębiorstwa, który został objęty modelowaniem i automatyzacją procesów biznesowych.

Lista opracowanych procesów biznesowych obejmuje m.in.:

- proces sprzedażowy,
- zapotrzebowanie na odzież,
- przyjęcie do magazynu,
- zamówienie do magazynu,
- inwentaryzację,
- fakturowanie (proces obsługi faktury wychodzącej),
- rejestrację dokumentu wchodzącego,
- rejestrację dokumentu wychodzącego,
- obsługę dokumentu kosztowego (proces obsługi faktury wchodzącej),
- zwolnienie pracownika,
- zatrudnienie pracownika,



RYS. 1. Zakres działań przedsiębiorstwa objęty automatyzacją procesów biznesowych
FIG. 1. The scope of company's activities covered by the automation of business processes

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

- zmianę szatni, szafy, skrytki,
- zmiany w obrębie odzieży (zmiana rozmiaru odzieży dla pracownika),
- zamówienie szafy/ akcesorium,
- zarejestrowanie odzieży własnościowej,
- wymianę akcesorium,
- zmianę danych osobowych,
- zmianę lokalizacji szafy,
- zmianę liczby sztuk,
- przeniesienie pracownika,
- zmiany szwalnicze w obrębie odzieży (naprawa odzieży),
- wykup asortymentu rentalowego.

Głównym założeniem usługi rentalu jest redukcja kosztów związanych z wielokrotnym, wręcz okresowym, zakupem tekstyliów i koniecznych nakładów na ich utrzymanie i obsługę oraz przekazanie odpowiedzialności w tym zakresie firmie zewnętrznej.

2. Wybrane procesy w serwisowaniu odzieży roboczej

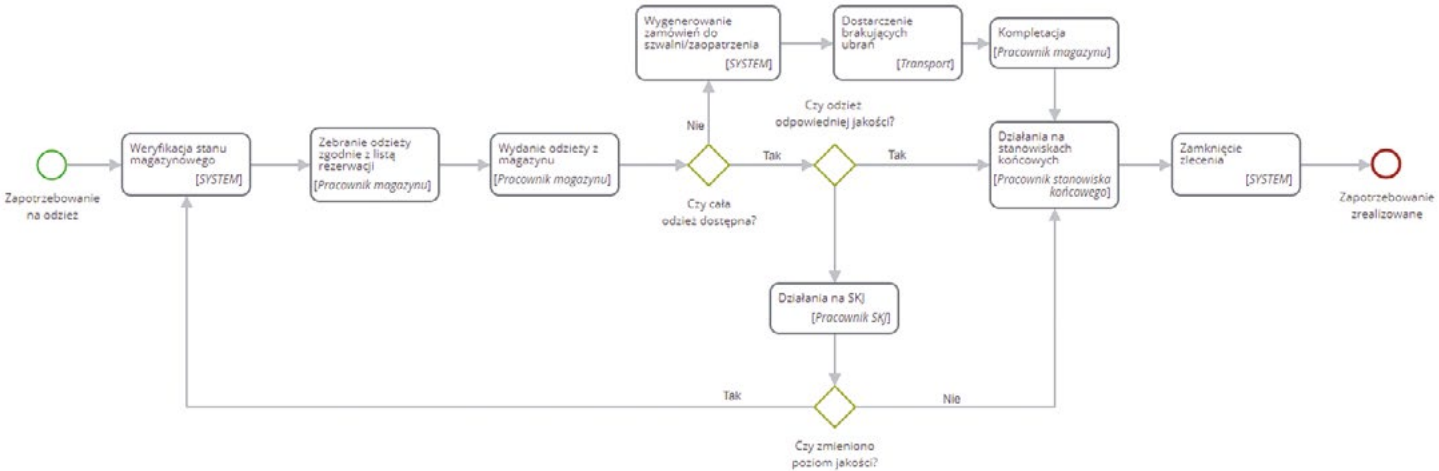
Dla ilustracji analizy w zakresie identyfikacji i badania własności procesów biznesowych firmy związanej z rentalem odzieży roboczej, wybrano z podanej już listy kilka charakterystycznych procesów w tym zakresie, a mianowicie:

- proces wydania odzieży z magazynu,
- proces zmiany statusu pojedynczej sztuki odzieży,
- proces obsługi zmian szwalniczych,
- proces zmiany parametrów odzieży,
- proces zmiany liczby sztuk odzieży.

W dalszej części opracowania przedstawiono je w postaci diagramów procesowych w standardzie BPMN.

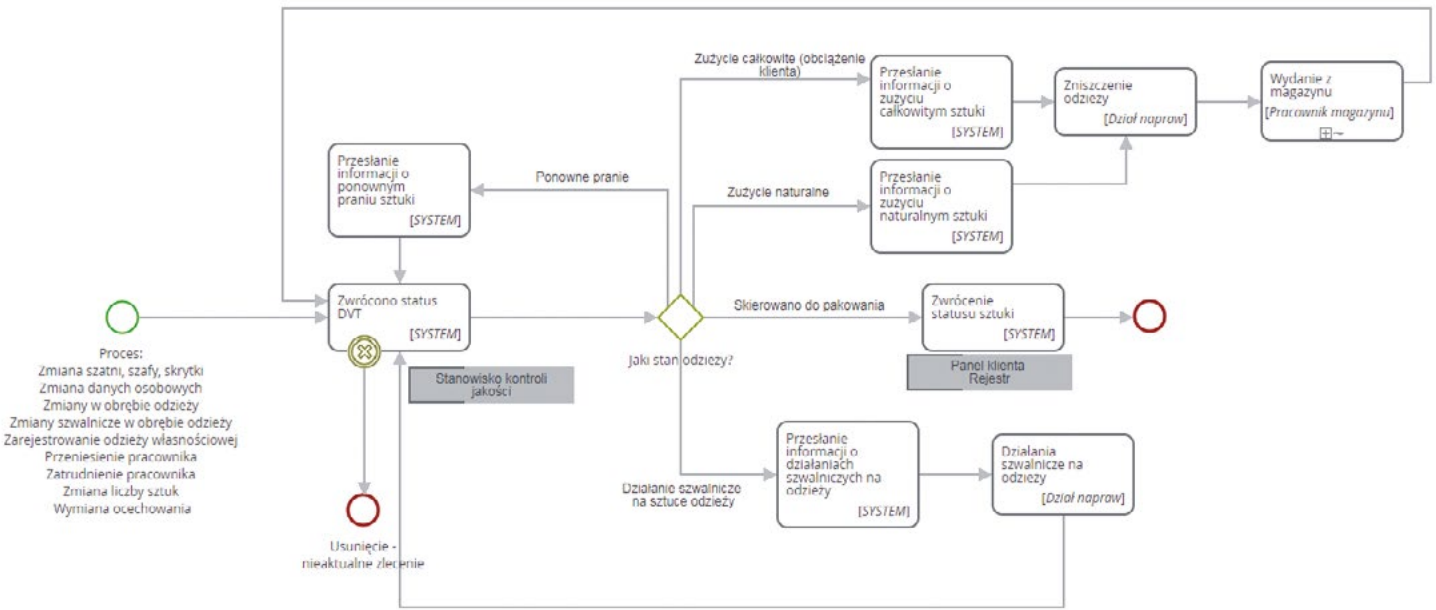
Proces wydania odzieży z magazynu ma na celu skompletowanie jej według zapotrzebowania i wydanie jej w określonym terminie z magazynu. Jest on realizowany przez pracowników magazynu zgodnie z obowiązującymi instrukcjami przy wykorzystaniu pulpitów dostępnych dla klienta poprzez przeglądarkę internetową. Proces rozpoczyna się od pojawienia się zapotrzebowania wynikającego z decyzji podjętych przez pracowników stanowiska kontroli jakości lub z działania procesów biznesowych związanych ze zgłoszeniami od klienta.

Proces swoim zakresem obejmuje wszystkie zadania magazyniera w zakresie prawidłowego wydania odzieży i został zobrazowany za pomocą przedstawionego diagramu na rysunku 2 i zaprojektowanego w systemie Aurea BPM [5].



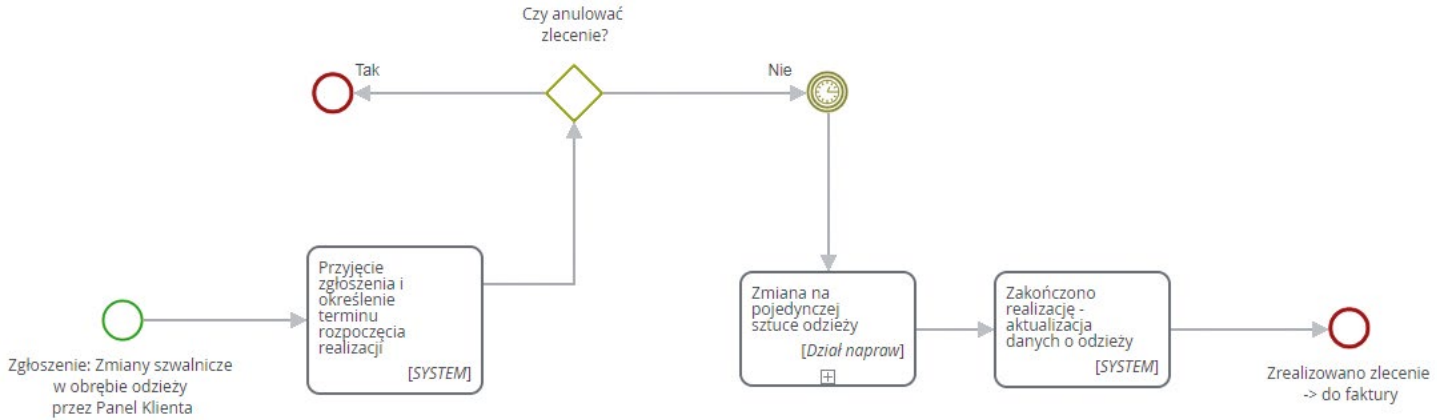
RYS. 2. **Proces wydania odzieży z magazynu w notacji BPMN**
FIG. 2. **Work clothes picking process BPMN model**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



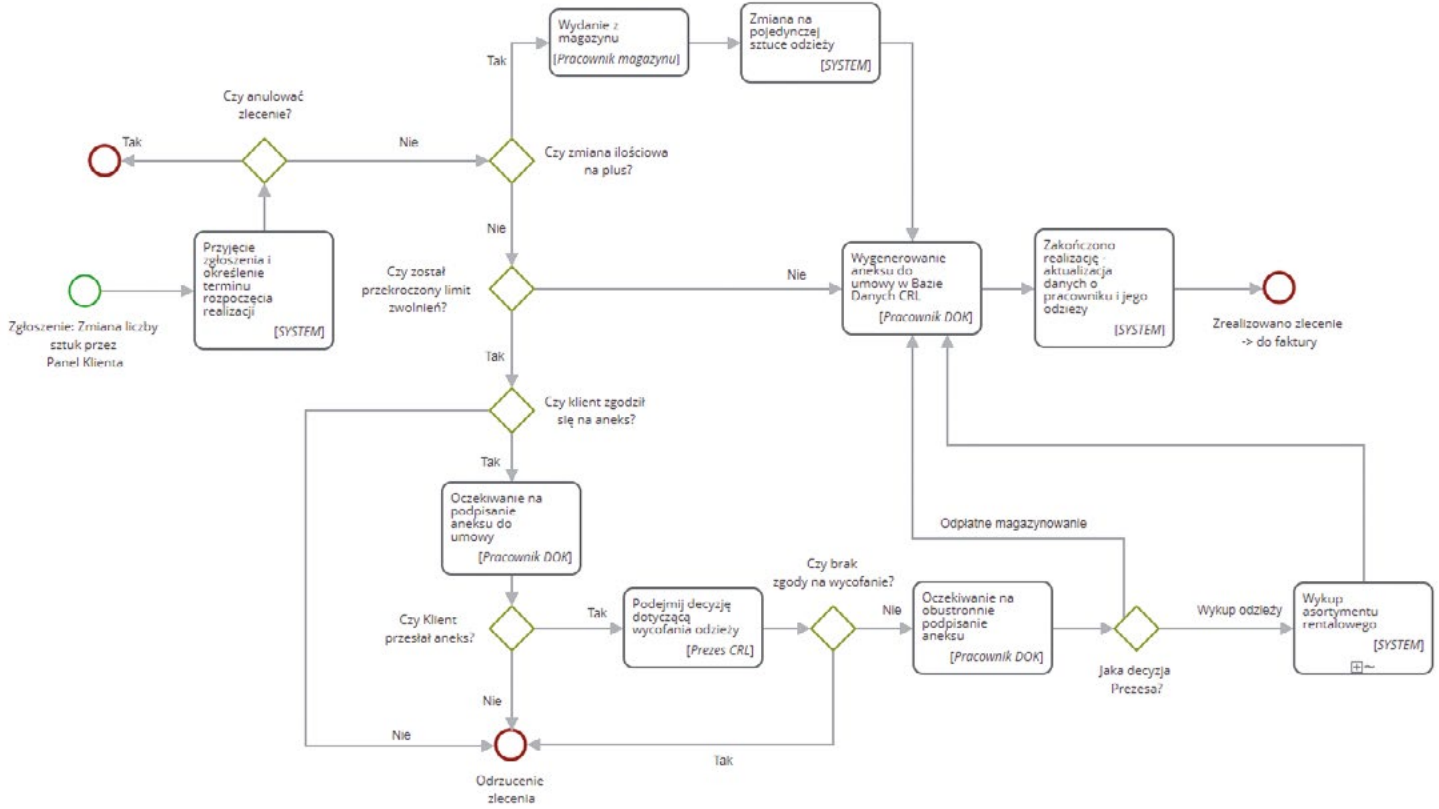
RYS. 3. **Proces zmiany statusu pojedynczej sztuki odzieży w notacji BPMN**
FIG. 3. **Status change of individual piece of clothing process BPMN model**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



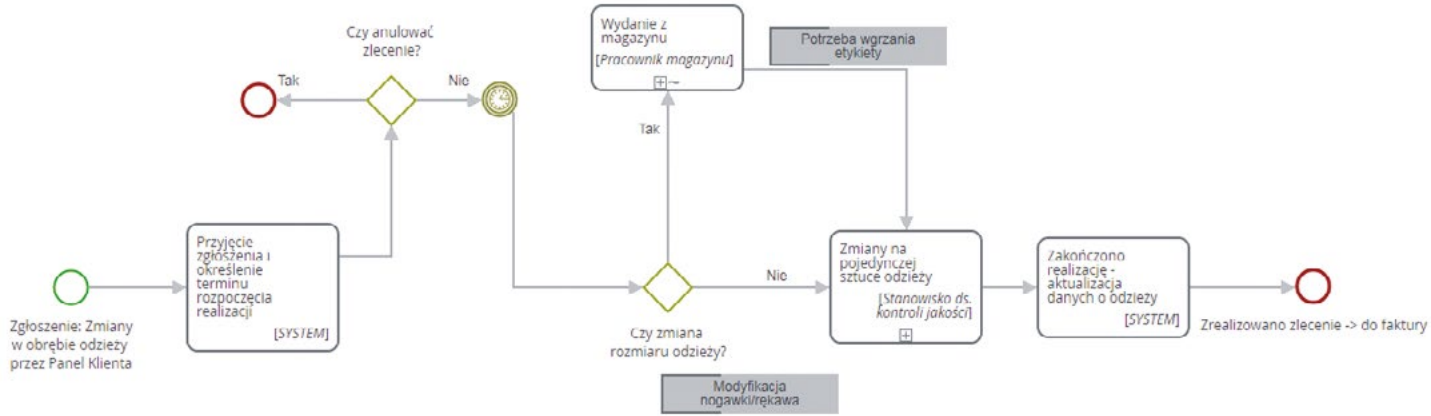
RYS. 4. **Proces obsługi zmian szwalniczych w notacji BPMN**
FIG. 4. **Sewing changes handling process BPMN model**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



RYS. 5. **Proces zmiany liczby sztuk odzieży w notacji BPMN**
FIG. 5. **Clothing amount change process BPMN model**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



RYS. 6. **Proces zmian parametrów odzieży w notacji BPMN**
FIG. 6. **Clothing parameters change process BPMN model**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Proces zmiany statusu pojedynczej sztuki odzieży (rys. 3) grupuje obsługę wszystkich zleceń od klienta, które mogą powodować konieczność wykonania pewnych zdefiniowanych czynności na pojedynczej sztuce odzieży.

Proces obsługi zmian szwalniczych (rys. 4) wspomaga obsługę zlecenia działań związanych z konkretną sztuką odzieży, które muszą być przeprowadzone w szwalni. Za jego pomocą wydawane są i odbierane zlecenia oraz planowana jest praca szwalni.

Proces zmiany liczby stuk odzieży (rys. 5) inicjowany jest na prośbę klienta, który, posługując się panelem klienta, zgłasza konieczność przygotowania dodatkowych kompletów odzieży dla określonej grupy pracowników. W ramach obsługi procesu podejmowanych jest szereg działań, które mają na celu przygotowanie i przekazanie zamówionych kompletów odzieży roboczej lub ochronnej.

Proces zmiany parametrów odzieży (rys. 6) wspomaga zbieranie zamówień oraz planowanie ich realizacji. Klient zgłasza zmiany w obrębie odzieży poprzez panel klienta dostępny z poziomu przeglądarki internetowej. Następnie planowane są i harmonogramowane odpowiednie działania mające na celu realizację zamówienia.

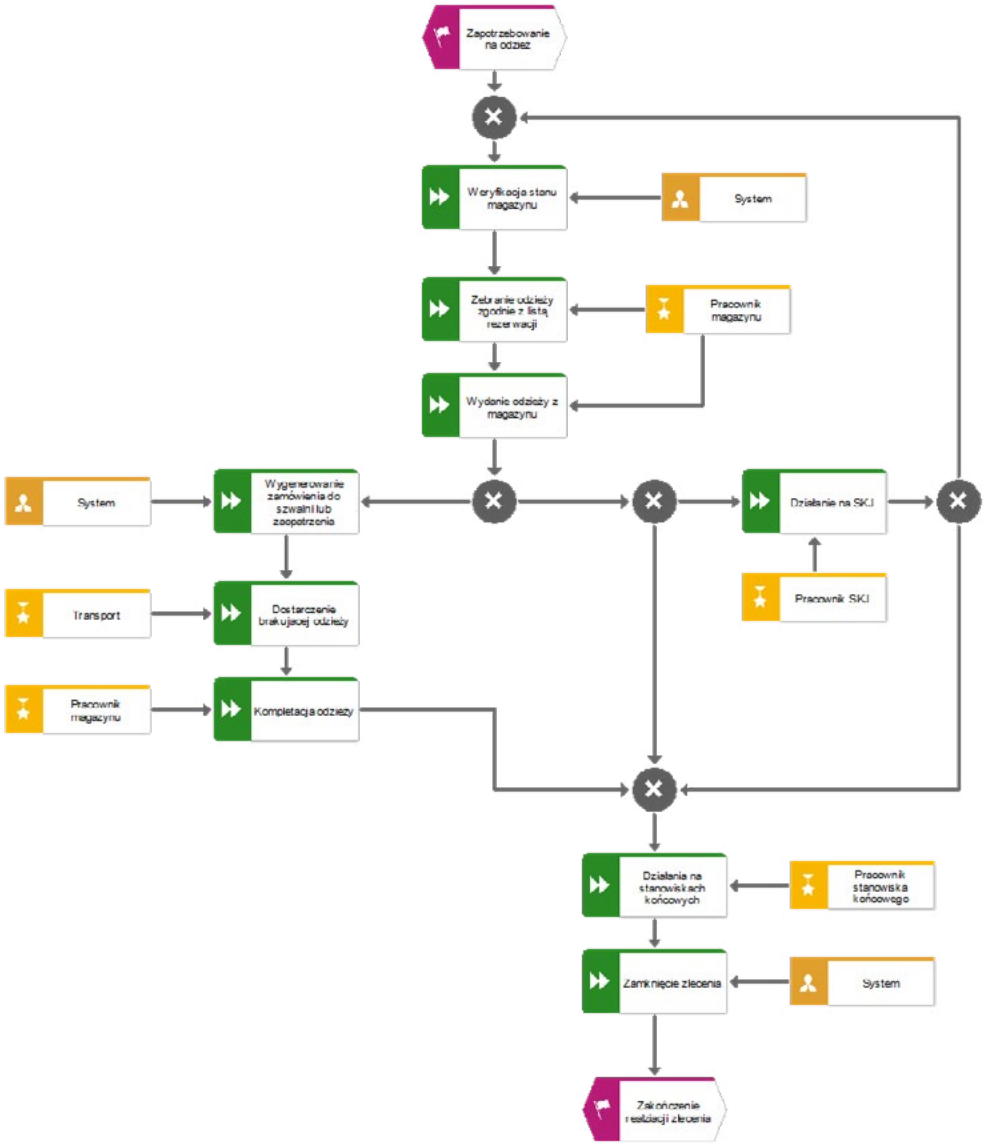
Wszystkie te procesy zostały zaimplementowane w środowisku wspomagania modelowania, symulacji i analizy procesów biznesowych ARIS 10 firmy Software AG. Przygotowane zostały ich wersje symulacyjne. Korzystając ze skonstruowanych w środowisku ARIS diagramów procesowych, opisano możliwości analizy dynamicznej tych procesów przedstawione w dalszej części.

3. Proces wydania odzieży z magazynu

Diagram procesu wydania odzieży z magazynu wykonany w środowisku ARIS ma postać przestawioną na rysunku 7.

Wprowadza się tu wykonawców poszczególnych czynności, takich jak pracownicy magazynu, stanowisk końcowych czy systemu kontroli jakości (SKJ), ale także system odzwierciedlający działanie systemu informatycznego firmy oraz transport, pokazujący funkcjonowanie własnych środków transportu. Nadaje się również losowe charakterystyki czasów i kosztów realizacji poszczególnych czynności i charakterystyki probabilistyczne przepływu sterowania w diagramie.

W efekcie tworzony jest model symulacyjny procesu, który skonstruowany w środowisku ARIS został pokazany na rysunku 8. Widać tu, że na bieżąco, w trakcie trwania eksperymentu symulacyjnego, zbierane i zobrazowywane są charakterystyki liczbowe różnych wielkości rejestrowanych w systemie.

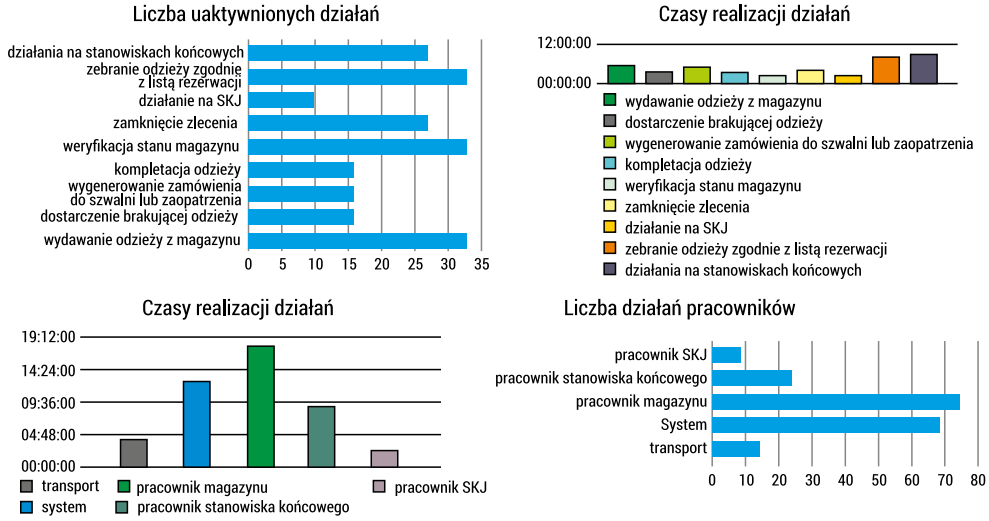


RYS. 7. **Proces wydania odzieży z magazynu**
FIG. 7. **Work clothes picking proces**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

RYS. 8. **Badanie symulacyjne procesu wydania odzieży z magazynu**
FIG. 8. **Simulation of the work clothes picking process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



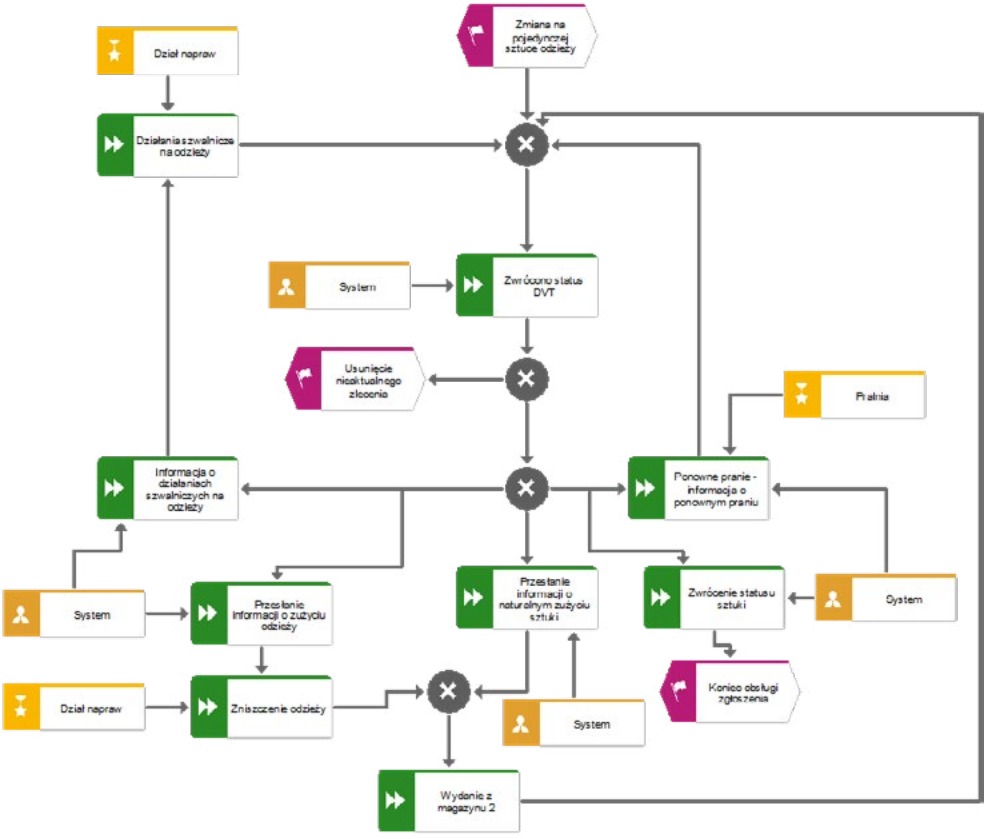
RYS. 9. **Wyniki badania symulacyjnego procesu wydania odzieży z magazynu**
FIG. 9. **Simulation results of the work clothes picking process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

Rysunek 9 ilustruje zestaw wybranych charakterystyk dynamicznych, jakie można uzyskać w efekcie wykonania serii eksperymentów symulacyjnych tygodniowego funkcjonowania magazynu firmy zajmującej się rentalem odzieży roboczej. Dotyczy to liczby wykonanych działań, zbiorczych czasów ich realizacji, sumarycznych godzin pracy poszczególnych pracowników, liczby podjętych przez pracowników poszczególnych typów działań różnego rodzaju itp. Dopiero symulacja funkcjonowania firmy pozwala na uzyskanie tego typu charakterystyk.

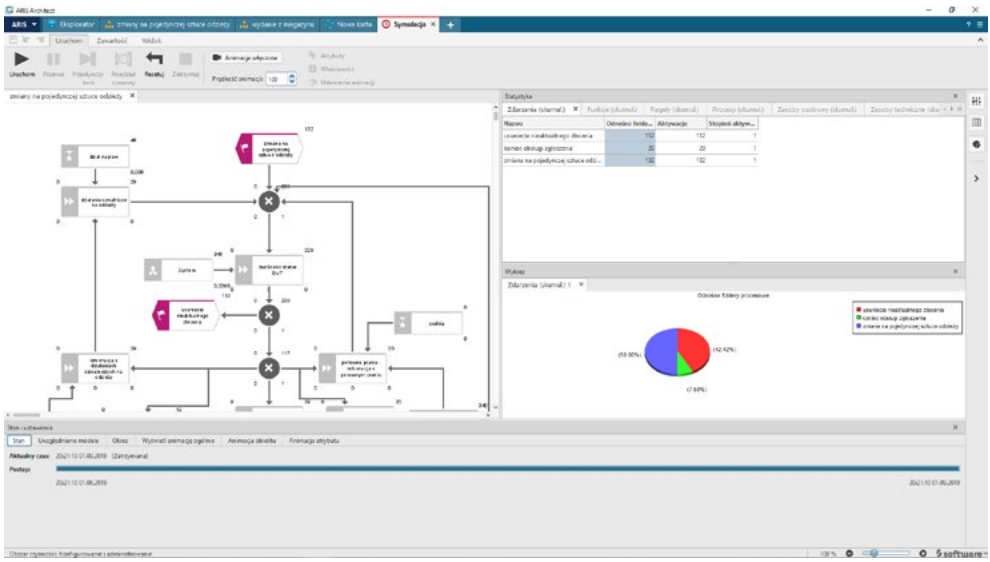
4. Proces zmiany pojedynczej sztuki odzieży

Diagram procesu zmiany pojedynczej sztuki odzieży wykonany w środowisku ARIS ma poniższą postać (rys. 10).



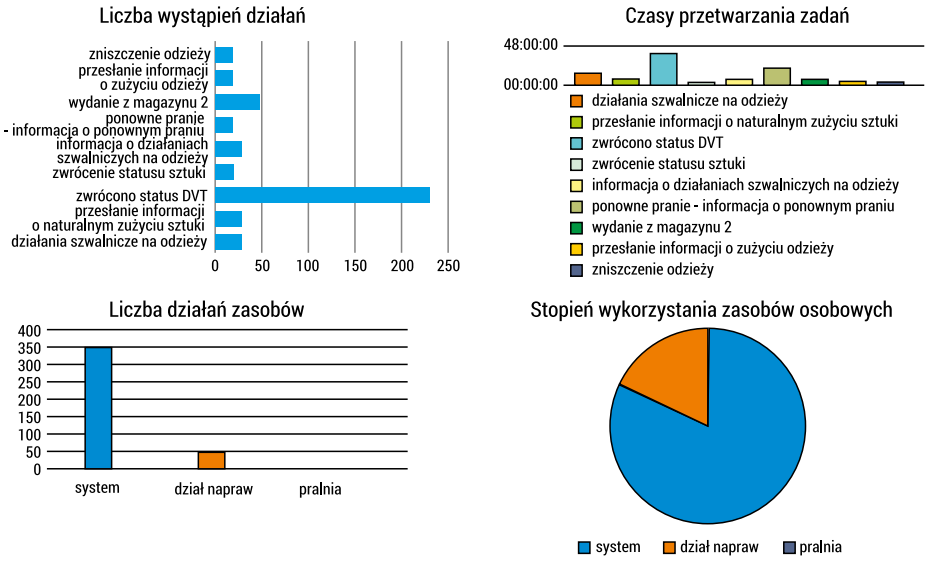
RYS. 10. **Proces zmiany statusu pojedynczej sztuki odzieży**
FIG. 10. **Status change of individual piece of clothing process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



RYS. 11. **Badanie symulacyjne procesu zmiany statusu pojedynczej sztuki odzieży**
FIG. 11. **Simulation of the status change of individual piece of clothing process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**



RYS. 12. **Wyniki badania symulacyjnego procesu wydania odzieży z magazynu**
FIG. 12. **Simulation results of the work clothes picking process**

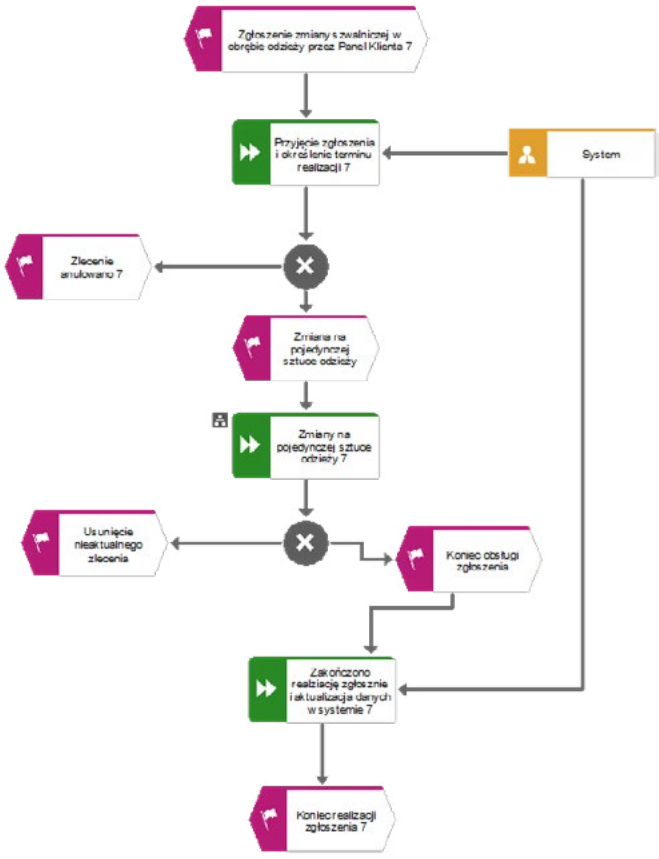
ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

W efekcie w środowisku ARIS tworzony jest model symulacyjny procesu, który został pokazany na rysunku 11.

Charakterystyki dynamiczne związane z liczbą wystąpień poszczególnych działań, sumarycznymi czasami tych działań, liczbą działań z użyciem zasobów poszczególnych typów, stopniem wykorzystania zasobów itp., zostały natomiast pokazane na rysunku 12.

5. Proces obsługi zmian szwalniczych

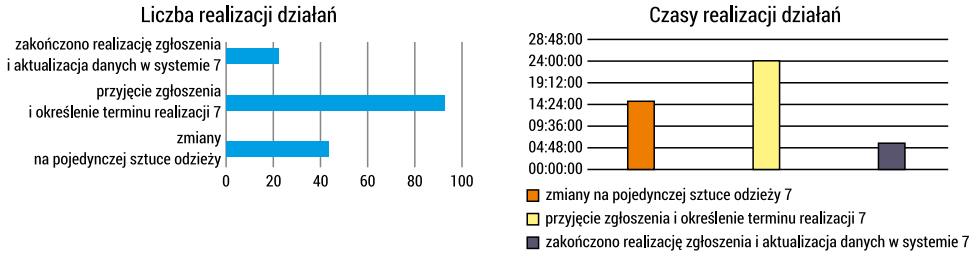
Diagram procesu obsługi zmian szwalniczych wykonany w środowisku ARIS ma poniższą postać (rys. 13):



RYS. 13. Proces obsługi zmian szwalniczych
FIG. 13. The process of handling sewing changes

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Charakterystyki związane z liczbą wystąpień poszczególnych działań i sumarycznymi czasami tych działań zostały natomiast pokazane na rysunku 14.



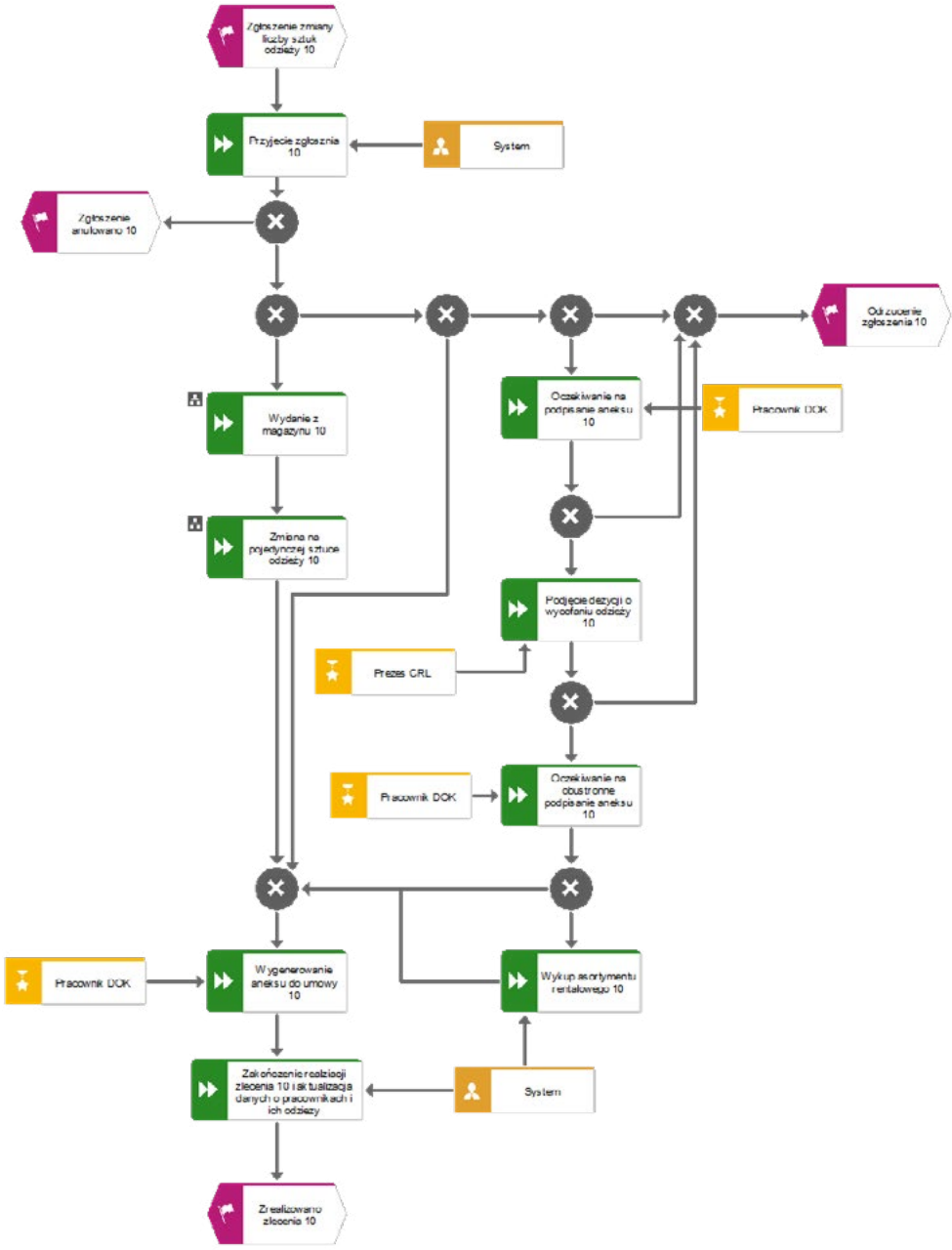
RYS. 14. **Wyniki badania symulacyjnego procesu zarządzania zmianami szwalniczymi**
FIG. 14. **Simulation results of the process of handling sewing changes**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

6. Proces zmiany liczby sztuk odzieży

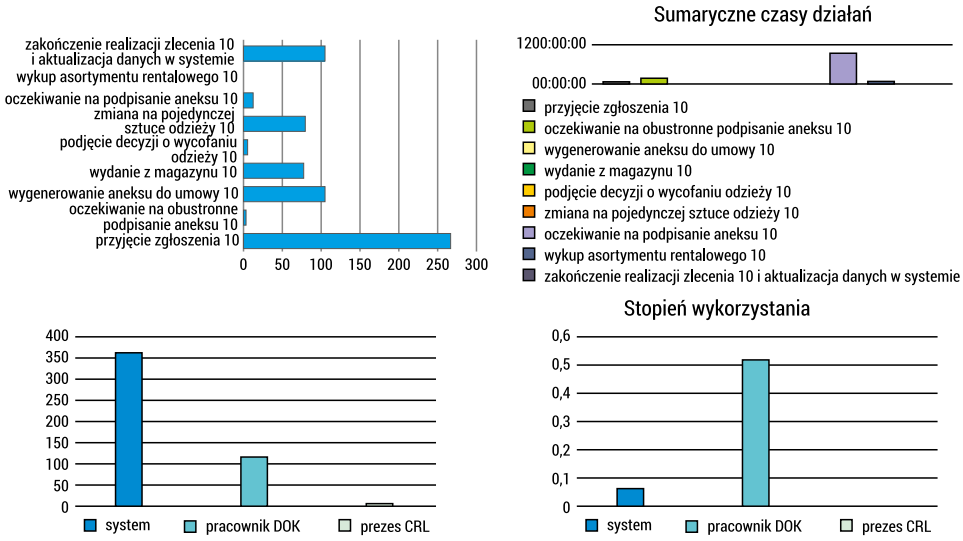
Diagram procesu zmiany liczby sztuk odzieży wykonany w środowisku ARIS ma poniższą postać (rys. 15).

Charakterystyki związane z liczbą zrealizowanych działań, sumarycznymi czasami tych działań, liczbą działań wykonanych przez poszczególne zasoby oraz stopniem wykorzystania poszczególnych zasobów zostały natomiast pokazane na rysunku 16.



RYS. 15. **Proces zmiany liczby sztuk odzieży**
FIG. 15. **Clothing amount change process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

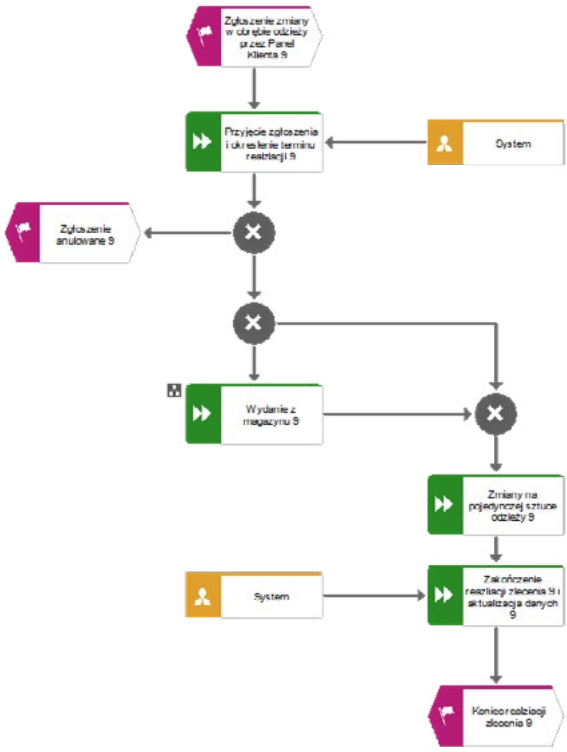


RYS. 16. **Wyniki badania symulacyjnego procesu zmiany liczy sztuk odzieży**
FIG. 16. **Simulation results of clothing amount change process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

7. Proces zmian parametrów odzieży

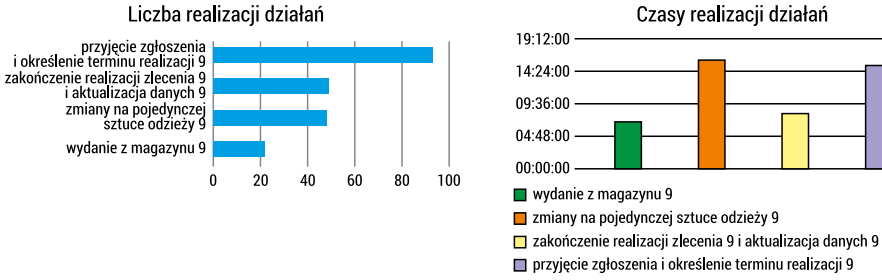
Diagram procesu zmiany zmian parametrów odzieży wykonany w środowisku ARIS ma poniższą postać (rys. 17):



RYS. 17. **Proces zmian parametrów odzieży**
FIG. 17. **Clothing parameters change process**

ŹRÓDŁO: **opracowanie własne**
SOURCE: **own elaboration.**

Charakterystyki związane z liczbą zrealizowanych działań oraz sumarycznymi czasami tych działań zostały natomiast pokazane na rysunku 18.



RYS. 18. **Wyniki badania symulacyjnego procesu zmian parametrów odzieży**
FIG. 18. **Simulation results of clothing parameters change process**

ŹRÓDŁO: **opracowanie własne.**
SOURCE: **own elaboration.**

8. Podsumowanie

Przedstawione powyżej modele stanowią bazę formalną do zastosowania zaawansowanych metod analizy wykonania, a także optymalizacji tychże działań.

Za pomocą notacji BPMN (w tym ARIS) daje się modelować wszystkie aspekty działań zgodnych z procedurami postępowania przyjętymi w przedsiębiorstwie w opraciu o najlepsze praktyki stosowane w budowie tego typu rozwiązań informatycznych. Poszczególne kroki procesu są zadaniami, które określone osoby lub komórki mają wykonać. Modele procesów biznesowych stanowią bazę formalną do zastosowania zaawansowanych metod prognozowania oraz podnoszenia skuteczności działań.

Literatura

1. Jasiulewicz-Kaczmarek M, Saniuk A, Nowicki T. The maintenance management in the macro-ergonomics context. In: Goossens RHM, eds *Advances in Social & Occupational Ergonomics Proceedings of the AHFE2016 Conference on Social & Occupational Ergonomics*, July 27-31, Walt Disney World®, Florida, USA Series: Advances in Intelligent Systems and Computing, vol. 487; 35-46; 2016.
2. Jasiulewicz-Kaczmarek M. The role of ergonomics in implementation of the social aspect of sustainability, illustrated with the example of maintenance. In: Arezes P, Baptista JS, Barroso M, Carneiro P, Lamb P, Costa N, Melo, R, Miguel AS, Perestrelo G, eds *Occupational Safety and Hygiene*. CRC Press, Taylor & Francis: London; 47-52; 2013.
3. Waszkowski R, Agata C, Kiedrowicz M, Nowicki T, Wesołowski Z Worwa K. *Data flow between RFID devices in a modern restricted access administrative office*, in 20th International Conference on Circuits, Systems, Communications and Computers, vol. 76, N. Mastorakis, V. Mladenov, and A. Bulucea, Eds. (MATEC Web of Conferences, Cedex A: E D P Sciences; 2016.
4. Waszkowski R, Kiedrowicz M, Nowicki T, Wesołowski Z, Worwa K. *Business processes in the RFID-equipped restricted access administrative office*, in 20th International Conference on Circuits, Systems, Communications and Computers, vol. 76, N. Mastorakis, V. Mladenov, and A. Bulucea, Eds. (MATEC Web of Conferences, Cedex A: E D P Sciences; 2016.
5. Aurea BPM system documentation (aurea-bpm.com).

Streszczenie

W pracy przedstawiono modele procesów biznesowych przedsiębiorstwa produkującego, wynajmującego i utrzymującego w stałym obrocie kilkaset tysięcy sztuk odzieży roboczej. Modele te opracowane zostały w celu usprawnienia efektywności działania przedsiębiorstwa.

Autorzy zaprezentowali również wyniki symulacyjnego badania własności opisanych wcześniej procesów biznesowych. Jako ilustracja możliwości opisanej metody użyte zostały wybrane procesy biznesowe automatyzujące pracę działu obsługi klienta.

Słowa kluczowe: zarządzanie procesami biznesowymi, BPM, modelowanie procesów biznesowych, platforma Low-code, efektywność, symulacja procesów, Aurea BPM

Summary
Simulation method for effectiveness evaluation in a working clothes rental company

The work presents models of business processes of a company that produces, washes and rents several hundred thousand pieces of work clothes. These models have been developed to improve efficiency of the company's operations.

The authors also presented results of a simulation evaluation of previously described business processes' properties. Selected processes in the area of customer service were used to illustrate effectiveness of the method being proposed.

Keywords: Business Process Management, Business Process Modeling, Low-code Platform, Efficiency, Process Simulation, Aurea BPM

DOI: 10.24427/978-83-66391-28-4_13

Sieci neuronowe w przetwarzaniu obrazów: przegląd wybranych osiągnięć

Karol PRZYBYSZEWSKI[*]

1. Wstęp

Od początku tego dziesięciolecia obserwujemy niepowstrzymywalny postęp, jaki dokonuje się w obszarze przetwarzania obrazów. Wzrost taniej mocy obliczeniowej oferowanej przez ogólnodostępne komputery oraz wzrost ilości dostępnych danych obrazowych umożliwiły szybsze eksperymentowanie i testowanie algorytmów. Spektakularnym przykładem sukcesu w tej dziedzinie są konwolucyjne sieci neuronowe, dla których pierwowzorem był neocognitron [1] przedstawiony przez Kunihiko Fukushimę w 1980 roku. Idea zaprezentowana przez Fukushimę była podstawą często cytowanej pracy Yann LeCun [2], w której została przedstawiona sieć LeNet-5, pionierska 7-poziomowa sieć konwolucyjna do klasyfikacji ręcznie pisanych cyfr. Była ona używana przez kilka banków do rozpoznawania ręcznie pisanych cyfr na zeskanowanych obrazach czeków o wielkości 32x32 piksele.

Nieprzerwany sukces konwolucyjnych sieci neuronowych jako efektywnej metody przetwarzania obrazów został ukoronowany sprzętową implementacją konwolucji na procesorach GPU [3]. Pozwoliło to na przyspieszenie obliczeń o co najmniej rząd wielkości i umożliwiło badaczom przeprowadzanie większej ilość eksperymentów w krótszym czasie. Obecnie najefektywniejsze architektury sieci neuronowych wykorzystywane do przetwarzania obrazów, takie jak: AlexNet (2012), ZFNet (2013), GoogleNet/Inception (2014), VGGNet (2014) czy ResNet (2015), oparte są o warstwy konwolucyjne. Historia powstania i rozwoju konwolucyjnych sieci neuronowych pokazuje, że od momentu odkrycia metody do momentu, kiedy w pełni zostanie wykorzystany jej potencjał, może minąć wiele lat.

W roku 2010 po raz pierwszy zostało zorganizowane wyzwanie IMAGENET Large Scale Visual Recognition Challenge (ILSVRC) [4], którego celem było „*oszacowanie zawartości zdjęć w celu ich pobrania i automatycznej adnotacji za pomocą podzbioru dużego, ręcznie oznakowanego zestawu danych ImageNet*". Wyzwanie to kontynuowane było przez osiem lat i w dużym stopniu przyczyniło się do szybkiego rozwoju technik przetwarzania obrazów. Prace publikowane w ramach ILSVRC pokazały

[*] Politechnika Białostocka

wyraźnie ten postęp – błąd klasyfikacji obrazów (top-5) został zredukowany z 0,28 (w 2010) do 0,023 (2017) [5]. Wyzwanie Imagenet przyczyniło się również do ponownego zainteresowania sieciami neuronowymi jako efektywnymi technikami analizy danych, a w szczególności danych obrazowych [6]. Wyzwanie ILSVRC zakończyło się w roku 2017 wskazaniem dalszych kierunków rozwoju technik przetwarzania danych – nastąpiło przesunięcie ciężaru badań z rozpoznawania/klasyfikacji obiektów do badań nad maszynowym 'rozumieniem' obrazu.

W opracowaniu przedstawiłem najważniejsze obszary dziedziny przetwarzania obrazów i różnych rodzajów i architektur sieci neuronowych zbudowanych w tym celu oraz opisałem wybrane koncepcje przetwarzania obrazów oparte o sieci neuronowe.

2. Dziedzina przetwarzania obrazów

Przetwarzanie obrazów, czasami nazywane też *widzeniem maszynowym* (od angielskiego *computer vision*) to pojemna dziedzina podzielona na wiele odrębnych obszarów. W celu jak najszerszego określenia i opisania dziedziny przetwarzania obrazów użyłem danych dostępnych i utrzymywanych przez ogólnodostępny serwis PapersWithCode (www.paperswithcode.com), którego misją jest prezentacja najnowszych osiągnięć z dziedziny uczenia maszynowego. W dziedzinie przetwarzania obrazów serwis ten zidentyfikował (maj 2019):

- 364 tablice wyników (ang. *leaderboards*),
- 501 zadania (ang. *tasks*),
- 173 ogólnodostępne zbiory danych (ang. *datasets*),
- 3524 artykuły naukowe z udostępnionym kodem źródłowym (ang. *papers with code*).

Zadania przetwarzania obrazów pogrupowane są w 219 obszarów, z których 20 najpopularniejszych (największa ilość artykułów) przedstawiłem w tabeli 1.

Strona PapersWithCode uwzględnia również mniej popularne czy nawet egzotyczne obszary, takie jak:

- *Pornography Detection,*
- *Damaged Building Detection,*
- *Logo Recognition,*
- *Window Detection,*
- *Transform A Video Into A Comics.*

TAB. 1. Wybrane obszary przetwarzania obrazów
TAB. 1. Selected areas of computer vision

Nr	Obszar	Ilość zadań	Dwa zadania z największą ilością artykułów
1.	Segmentacja semantyczna (ang. *Semantic Segmentation*)	7	*Semantic Segmentation* – 417 art. *Real-Time Semantic Segmentation* – 22 art.
2.	Klasyfikacja obrazów (ang. *Image Classification*)	7	*Image Classification* – 353 art. *Few-Shot Image Classification* – 14 art.
3.	Detekcja obiektów (ang. *Object Detection*)	17	*Object Detection* – 298 art. *3D Object Detection* – 20 art.
4.	Odpowiadanie na pytania (ang. *Question Answering*)	6	*Question Answering* – 283 art. *Open-Domain Question Answering* – 15 art.
5.	Generowanie obrazów (ang. *Image Generation*)	8	*Image generation* – 158 art. *Image-to-Image translation* – 61 art.
6.	Określanie pozy sylwetki (ang. *Pose Estimation*)	9	*Pose Estimation* – 146 art. *3D Human Pose Estimation* – 25 art.
7.	Zwiększanie rozdzielczości (ang. *Super Resolution*)	4	*Super Resolution* – 124 art. *Image Super-Resolution* – 77 art.
8.	Pojazdy autonomiczne (ang. *Autonomous Vehicles*)	13	*Autonomous Driving* – 84 art. *Autonomous Vehicles* – 38 art.
9.	Rozpoznawanie i modelowanie twarzy (ang. *Facial Recognition and Modelling*)	30	*Face Recognition* – 65 art. *Face Detection* – 37 art.
10.	Obraz wideo (ang. *Video*)	35	*Object tracking* – 48 art. *Video Classification* – 30 art.
11.	Rozpoznawanie obiektów (ang. *Object Recognition*)	4	*Object Recognition* – 113 art. *3D Object Recognition* – 7 art.
12.	Wyszukiwanie obrazem (ang. *Image Retrieval*)	7	*Image Retrieval* – 105 art. *Content-Based Image Retrieval* – 9 art.
13.	Rozpoznawanie akcji (ang. *Action Recognition*)	6	*Action Recognition* – 89 art. *Action Recognition in Videos* – 13 art.
14.	Tagowanie obrazów (ang. *Image Captioning*)	2	*Image Captioning* – 96 art. *Image Paragraph Captioning* – 1 art.

Nr	Obszar	Ilość zadań	Dwa zadania z największą ilością artykułów
15.	Określanie głębokości (ang. *Depth Estimation*)	9	*Depth Estimation* – 68 art. *Monocular Depth Estimation* – 24 art.
16.	Transfer stylu (ang. *Style Transfer*)	5	*Style Transfer* – 86 art. *Image Stylization* – 7 art.
17.	Detekcja anomalii (ang. *Anomaly Detection*)	7	*Anomaly Detection* – 81 art. *Unsupervised Anomaly Detection* – 8 art.
18.	Rozumienie sceny (ang. *Scene Parsing*)	8	*Scene Understanding* – 40 art. *Scene Recognition* – 15 art.
19.	Obrazy 3D (ang. *3D*)	25	*3D Reconstruction* – 47 art. *3D Pose estimation* – 17 art.
20.	Ponowna identyfikacja osoby (ang. *Person Re-Identification*)	6	*Person Re-Identification* – 66 art. *Video-Based Person Re-Identification* – 4 art.

Źródło: opracowanie własne.
SOURCE: own elaboration.

Obszary przetwarzania obrazów wymienione w powyższej tabeli jasno pokazują że ta dziedzina ma zastosowanie w wielu praktycznych aspektach. Najczęściej używane techniki to segmentacja/detekcja oraz klasyfikacja. Istotnym zagadnieniem jest również łączenie technik przetwarzania obrazów z technikami przetwarzania języka naturalnego w takich obszarach, jak Odpowiadanie na pytania, Tagowanie obrazów czy Rozumienie Sceny. Coraz bardziej popularne są również techniki związane z tworzeniem nowych oraz modyfikacją istniejących obrazów, takie jak Generowanie Obrazów czy też Transfer stylu. Warto również zwrócić uwagę na to, że dziedzina przetwarzania obrazów nie ogranicza się jedynie do obrazów dwuwymiarowych, ale porusza również zagadnienia związane z nagraniami wideo oraz obrazami trójwymiarowymi.

3. Sieci neuronowe w przetwarzaniu obrazów

Niekwestionowanym liderem w dziedzinie przetwarzania obrazów jest koncepcja konwolucyjnych sieci neuronowych (ang. *convolutional neural networks*), czasami też tłumaczona na język polski jako *splotowe sieci neuronowe*. Ich ogromna zaleta to zdolność redukcji obrazu do formy, która jest o wiele prostsza do przetwarzania, ale zachowuje wszystkie cechy istotne do poprawnego wnioskowania. Istnieje jednak wiele innych koncepcji związanych z sieciami neuronowymi, które również dają

bardzo dobre rezultaty w obszarze przetwarzania obrazów. W poniższej tabeli wymienione zostały wszystkie istotne koncepcje związane z sieciami neuronowymi, jakie dostępne były w czasie pisania artykułu. Do opisu użyłem nazw anglojęzycznych, gdyż w wielu przypadkach nie wykształciła się jeszcze terminologia polska i dla zachowania spójności nazewniczej lepiej jest użyć jednolitego, angielskiego nazewnictwo.

TAB. 2. Istotne koncepcje sieci neuronowych związane z przetwarzaniem obrazów
TAB. 2. Important neural networks concepts related to computer vision

Nr	Nazwa	Rok publikacji
1.	Feed forward neural networks (FF or FFNN) i perceptrons (P)	1958
2.	Hopfield network (HN)	1982
3.	Kohonen networks (KN, także Self Organising (Feature) Map, SOM, SOFM)	1982
4.	Boltzmann machines (BM)	1986
5.	Restricted Boltzmann machines (RBM)	1986
6.	Radial basis function (RBF)	1988
7.	Autoencoders (AE)	1988
8.	Recurrent neural networks (RNN)	1990
9.	Long / short term memory (LSTM)	1997
10.	Bidirectional recurrent neural networks, bidirectional long / short term memory networks i bidirectional gated recurrent units (BiRNN, BiLSTM i BiGRU odpowiednio)	1997
11.	Convolutional neural networks (CNN lub deep convolutional neural networks, DCNN)	1998
12.	Liquid state machines (LSM)	2002
13.	Echo state networks (ESN)	2004
14.	Extreme learning machines (ELM)	2006
15.	Sparse autoencoders (SAE)	2007
16.	Deep belief networks (DBN)	2007

Nr	Nazwa	Rok publikacji
17.	Denoising autoencoders (DAE)	2008
18.	Deconvolutional networks (DN), również nazywane Inverse Graphics Networks (IGNs)	2010
19.	Variational autoencoders (VAE)	2013
20.	Markov chains (MC lub discrete time Markov Chain, DTMC)	2013
21.	Gated recurrent units (GRU)	2014
22.	Generative adversarial networks (GAN)	2014
23.	Neural Turing machines (NTM)	2014
24.	Deep convolutional inverse graphics networks (DCIGN)	2015
25.	Deep residual networks (DRN)	2015
26.	Attention networks (AN)	2015
27.	Differentiable Neural Computers (DNC)	2016
28.	Capsule Networks (CapsNet)	2017

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Powyższa tabela wyraźnie pokazuje aktywny rozwój architektur i koncepcji związanych z sieciami neuronowymi. Koncepcje te rozwijane są na różnych poziomach, przykładowo:

- sieci CapsNet skupiają się w dużym stopniu na zlikwidowaniu pewnych ograniczeń sieci konwolucyjnych i usprawnieniu ich zdolności bardziej efektywnej ekstrakcji istotnych cech;
- w koncepcjach, takich jak GAN czy autoenkodery, istotnym elementem jest odpowiednia kombinacja współpracujących ze sobą sieci neuronowych.

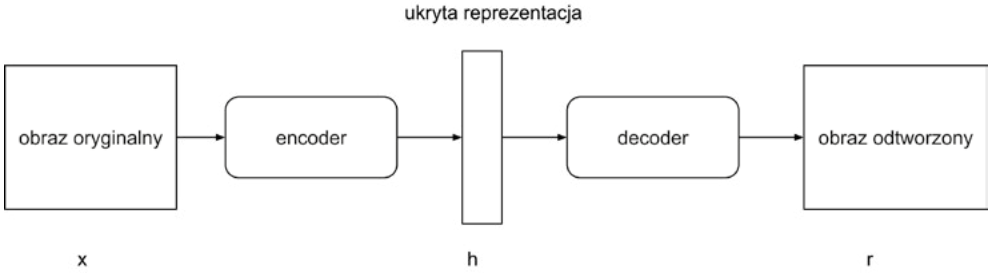
4. Koncepcje autoenkodera (AE) i GAN

W tym paragrafie przedstawione zostały koncepcja GAN [7] i koncepcja autoenkoderów, które wychodzą poza zwyczajowe zastosowanie sieci neuronowych, czyli klasyfikację, a ich zastosowanie w przetwarzaniu obrazów dało spektakularne rezultaty. GAN-y i autoenkodery są stosowane do generowania, modyfikacji czy też edycji obrazów.

Autoenkodery

Idea autoenkoderów już od wielu lat jest częścią obszaru zagadnień związanych z sieciami neuronowymi, jednak dopiero niedawne zastosowanie w przetwarzaniu obrazów dało spektakularne rezultaty, np. w postaci transferu wybranego stylu malarskiego na dowolne, amatorskie zdjęcia wykonane zwykłym aparatem.

Głównym założeniem autoenkoderów jest redukcja danych wejściowych do ukrytej przestrzeni stanów o mniejszej liczbie wymiarów, a następnie próba odtworzenia danych wejściowych z tej reprezentacji. Pierwsza część nazywa się kodowaniem, a druga – fazą dekodowania. Zmniejszając liczbę zmiennych reprezentujących dane, wymuszamy na modelu nauczenie się, jak zachować tylko istotne informacje, z których dane wejściowe można odtworzyć. Działanie to może być również postrzegane jako technika kompresji.



RYS. 1. Architektura autoenkodera

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Enkoder ma postać $h = f(x)$, natomiast dekoder: $r = g(h)$. Tradycyjnie, autoenkodery minimalizują funkcję:

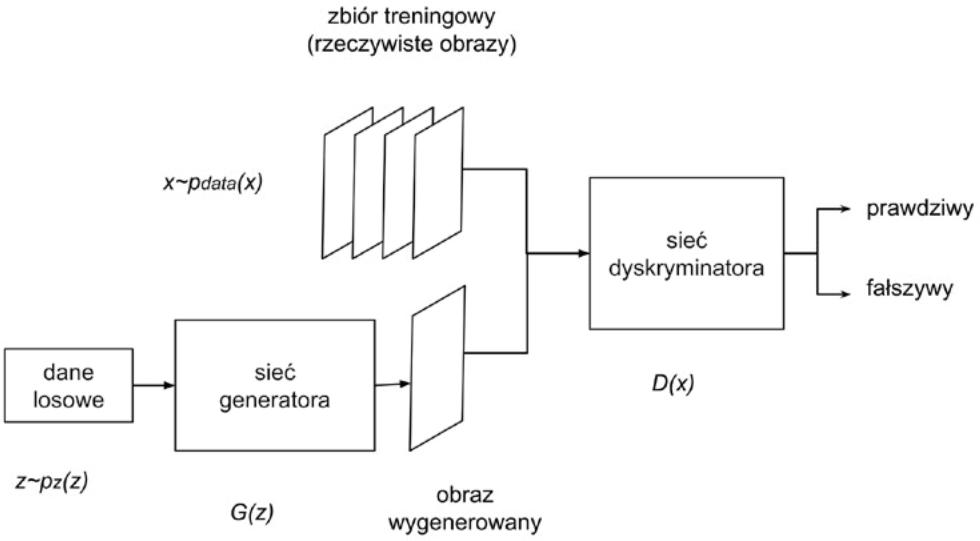
$$L(x,g(f(x)))$$

gdzie L jest funkcją kosztu karzącą $g(f(x))$ gdy nie jest ona podobna do x (przykładowo normę L^2 z ich różnicy). Koncepcja autoenkoderów okazała się bardzo pojemna, a w jej ramach można przykładowo wyróżnić:

- *Undercomplete Autoencoders* – gdzie rozmiar *h* jest mniejszy od *x* (tradycyjna, podstawowa wersja autoenkodera);
- *Sparse Autoencoders* (*SAE*) [8] – do funkcji kary dodawany jest parametr karzący za nierzadką reprezentację; *SAE* zwykle są używane do ekstrakcji cech na potrzeby innych zadań, takich jak klasyfikacja;
- *Denoising Autoencoders* (*DAE*) [9] – jako wejście otrzymują uszkodzone dane (np. zaszumiony obraz) i trenowane są w celu otrzymania oryginalnych, nieuszkodzonych danych jako wyjścia;
- *Variational Autoencoders* (*VAE*) [10] – dodaje człon regularyzujący, wymuszający odpowiedni rozkład sygnału w warstwie kodującej; w przetwarzaniu obrazów VAE używane mogą być do generowania nowych obrazów;
- oraz inne, niewymienione w tym opracowaniu.

Generative adversarial networks (GAN)

Rozpatrując koncepcję GAN z wysokiego poziomu, można stwierdzić, że architektura sieci GAN składa się z dwóch komponentów: generatora i dyskryminatora. Dyskryminator ma za zadanie określić, czy dany obraz wygląda naturalnie (to znaczy, czy jest obrazem z zestawu danych) lub czy wygląda na sztucznie utworzony. Zadaniem generatora jest natomiast tworzenie naturalnie wyglądających obrazów, które są podobne do pierwotnego rozkładu danych, czyli obrazów które wyglądają na tyle naturalnie by oszukać sieć dyskryminatora.



RYS. 2. Architektura koncepcji GAN

ŹRÓDŁO: opracowanie własne.
SOURCE: own elaboration.

Notacja przedstawiona na rysunku oznacza: $p_{data}(x)$ – rozkład prawdopodobieństwa rzeczywistych obrazów, x – próbka z rozkładu $p_{data}(x)$, $p_z(z)$ – rozkład prawdopodobieństwa generatora, z – próbka z rozkładu $p_z(z)$, $G(z)$ – sieć generatora, $D(x)$ – sieć dyskryminatora. Tak jak wspominałem wcześniej, trening sieci GAN jest realizowany jako rywalizacja między generatorem a dyskryminatorem. Można to opisać matematycznie jako:

$$min_G\, max_D\, V(D, G) = \mathbb{E}_{x\sim pdata(x)}[log(D(x))] + \mathbb{E}_{z\sim pz(z)}[log(1 - D(G(z))] \qquad (2)$$

W funkcji $V(D, G)$ pierwszym członem jest entropia pokazująca, że dane z prawdziwego rozkładu ($p_{data}(x)$) przejdą przez dyskryminator (najlepszy scenariusz). Dyskryminator stara się maksymalizować tą wartość do 1. Drugi człon to entropia pokazująca, że dane z losowego rozkładu ($p_z(z)$) przechodzą przez generator wytwarzający nieprawdziwy obraz, który jest następnie przepuszczany przez dyskryminator w celu identyfikacji autentyczności (najgorszy przypadek). Ten człon dyskryminator stara się doprowadzić do wartości 0. Całościowo zatem dyskryminator dąży do maksymalizacji funkcji V. Z drugiej strony zadanie generatora jest dokładnie odwrotne. Stara się on minimalizować funkcję V w taki sposób, aby różnica między prawdziwymi a wytworzonymi danymi była jak najmniejsza. W terminologii angielskiej do opisu tej koncepcji używa się terminu *minmax game*. Należy również wspomnieć, że koncepcja GAN ma wiele różnych zastosowań i odmian [11].

AE versus GAN

Autoenkoder kompresuje swoje dane wejściowe do wektora o znacznie mniejszych wymiarach niż dane wejściowe, a następnie przekształca je z powrotem w wektor o tym samym kształcie. GAN można opisać jako odwrócony autoenkoder – zamiast kompresować dane wielowymiarowe, dostaje wektory niskowymiarowe jako dane wejściowe, a dane o dużych wymiarach znajdują się w środku architektury sieci.

GAN nie przyjmuje rzeczywistych danych jako wejściowych, a zamiast tego otrzymuje mały wektor liczb losowych. Sieć generatora próbuje przekształcić ten mały wektor w realistyczną próbkę z danych treningowych. Następnie sieć dyskryminatora pobiera tę wygenerowaną próbkę (i kilka rzeczywistych próbek z zestawu danych) i uczy się odgadywać, czy próbki są prawdziwe czy fałszywe.

5. Sieci kapsułkowe

Konwolucyjne sieci neuronowe (CNN) odniosły wielki sukces w rozwiązywaniu problemów z rozpoznawaniem obiektów i ich klasyfikacją. Nie są jednak idealne. Jeśli na wejście sieci konwolucyjnej podamy obiekt w orientacji, której sieć nie zna lub w której obiekty pojawiają się w miejscach, do których sieć nie jest przyzwyczajona,

zadanie predykcji prawdopodobnie się nie powiedzie. CNN uczą się wzorów statystycznych na obrazach, ale nie uczą się podstawowych pojęć dotyczących tego, co sprawia, że coś rzeczywiście wygląda jak konkretny rzeczywisty obiekt (np. twarz).

W 2017 Geoffrey Hinton (i inni), zapożyczyli pomysły z neurobiologii, które sugerują, że mózg jest zorganizowany w moduły zwane kapsułkami [12] (*CapsNets*). Kapsułki te są szczególnie dobre w rozpoznawaniu cech takich, jak ułożenie (położenie, rozmiar, orientacja), deformacja, prędkość, albedo, odcień, tekstura itp. W kontekście sieci neuronowych kapsułki reprezentowane są przez grupy neuronów.

Rezultaty zaprezentowane w pracach Hintona pokazały, że *CapsNets* mają najwyższą wydajność w standardowych zestawach danych, takich jak MNIST (z dokładnością testową 99,75%) i SmallNORB (z 45% zmniejszeniem błędu w stosunku do poprzedniego najlepszego wyniku). Jednak aplikacje i wydajność tych sieci na rzeczywistych i bardziej złożonych danych nie zostały w pełni zweryfikowane. Bardzo ważną korzyścią, jaką zapewniają sieci kapsułkowe, jest przejście od sieci neuronowych typu *black-box* do tych, które reprezentują bardziej konkretne cechy, mogące pomóc nam przeanalizować i zrozumieć, w jaki sposób sieć neuronowa działa od środka. Należy również wspomnieć, że *CapsNets* używają koncepcji autoenkoderów (rekonstrukcyjna funkcja kosztu jest używana jako metoda regularyzacji), co potwierdza tezę, że idee opracowane lata temu wciąż znajdują ważne zastosowania i przyczyniają się do powstawania nowych, efektywnych rozwiązań w różnych dziedzinach eksploracji danych.

6. Powiązane prace

Gwałtowny rozwój przetwarzania obrazów przy użyciu sieci neuronowych zaowocował również dużą ilością artykułów naukowych opisujących przekrojowo wybrane zagadnienia z tej dziedziny. Warto zwrócić uwagę na artykuł [13], w którym autorzy obszernie opisują ponad 20 lat historii wykrywania obiektów. Artykuł oparty jest o przegląd ponad 400 prac obejmujących okres od 1990 do 2019 roku. Autorzy wyraźnie pokazali podział na dwie główne epoki detekcji obiektów: tradycyjne metody detekcji (do 2012) i metody oparte o głębokie uczenie maszynowe (po 2012), wśród których najpopularniejsze są koncepcje związane z sieciami konwolucyjnymi.

Bardzo dobrą i obszerną pracą opisującą współczesne architektury głębokiego uczenia maszynowego jest artykuł [14]. Autorzy kompleksowo opisują rozwój najważniejszych koncepcji w dziedzinie głębokiego uczenia maszynowego od roku 2012. Do artykułu dołączona jest również lista najpopularniejszych frameworków, SDK oraz referencyjnych zbiorów danych używanych do implementacji i ewaluacji zadań związanych z głębokim uczeniem maszynowym.

Warto również zwrócić uwagę na [15,] która skupia się na opisie historii rozwoju głębokich konwolucyjnych sieci neuronowych. Badanie skupia się na pokazaniu wewnętrznej taksonomii najnowszych, głębokich architektur CNN. Podejmuje też próbę podziału najnowszych innowacji w architekturach CNN na siedem różnych kategorii (terminologia

angielska): *spatial exploitation, depth, multi-path, width, feature map exploitation, channel boosting* oraz *attention*. Wartościowych informacji dostarcza dołączona na końcu artykułu tabela porównująca wyniki różnych architektur w odniesieniu do wyżej wymienionych kategorii.

7. Wnioski i dalsze prace

Na dzień pisania pracy nie znalazłem opracowań naukowych, które całościowo podejmują temat wysokopoziomowej taksonomii architektur i koncepcji opartych o sieci neuronowe, wraz ze wskazaniem i kategoryzacją praktycznych obszarów badawczych przetwarzania obrazów. Niniejsze opracowanie jest wstępem do dalszych prac nad opisaniem rozwoju architektur, koncepcji i modeli sieci neuronowych w oparciu o ich praktyczne zastosowania.

Lista praktycznych obszarów badawczych zaprezentowana w pracy pokazuje, że współczesne przetwarzanie obrazów to dziedzina szeroka i odważnie wkraczająca na nowe pola badań. Krótka historia kluczowych odkryć w obszarze sieci neuronowych pozwala zauważyć, że od momentu opracowania podstawowych koncepcji (tutaj: neuron) do czasu kiedy to odkrycie (tutaj: sieci konwolucyjne, autoenkodery, GAN-y, etc) osiągnie wysoki potencjał, mogą minąć dziesięciolecia.

Niniejsze opracowanie jest pierwszym z planowanej serii opisującej wysokopoziomowe koncepcje i architektury sieci neuronowych oraz łączenie tej wiedzy z praktycznymi obszarami zastosowań technik przetwarzania obrazów. W ramach dalszych prac będę starał odpowiedzieć się na takie pytania, jak:

- Jak przeprowadzić dalszą analizę struktury/architektury sieci versus zastosowania praktyczne?
- Jak szukać odpowiednich architektur dla konkretnych zastosowań?
- Jak budować nowe architektury?
- Jak skategoryzować koncepcje w ramach sieci neuronowych?

Literatura

1. Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybernetics*. 1980. 36: 193-202.
2. Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. 1998; vol. 86, no. 11: 2278-2324; DOI: 10.1109/5.726791.
3. Chellapilla K, Puri S, Simard P. *High Performance Convolutional Neural Networks for Document Processing. Tenth International Workshop on Frontiers in Handwriting Recognition*, Université de Rennes 1, Oct 2006, La Baule (France). ffinria-00112631.

4. http://image-net.org/challenges/LSVRC/2010/index.
5. http://image-net.org/challenges/talks_2017/ILSVRC2017_overview.pdf.
6. Krizhevsky A, Sutskever I, Hinton GE. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84-90. DOI: https://doi.org/10.1145/306538.
7. Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. *Generative Adversarial Networks*, 2014. Conference: Advances in neural information processing systems; 2672-2680.
8. Hinton GE, Osindero S, The Y-W. 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18, 7 (July 2006), 1527-1554. DOI:https://doi.org/10.1162/neco.2006.18.7.1527.
9. Vincent P, Larochelle H, Bengio Y, Manzagol P-A. 2008. Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning (ICML '08). Association for Computing Machinery, New York, NY, USA, 1096-1103. DOI:https://doi.org/10.1145/1390156.1390294.
10. Kingma DP, Welling M. *Auto-Encoding Variational Bayes*. Paper presented at the meeting of the ICLR, 2014.
11. Lucic M, Kurach K, Michalski M, Bousquet O, Gelly S. 2018. Are GANs created equal? a large-scale study. In Proceedings of the 32nd International Conference on Neural Information Processing Systems (NIPS'18). Curran Associates Inc., Red Hook, NY, USA; 698-707.
12. Sabour S, Frosst N, Hinton GE. 2017. Dynamic routing between capsules. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 3859-3869.
13. Zhengxia Z, Shi Z, Guo Y, Ye J. "Object detection in 20 years: A survey". arXiv preprint arXiv:1905.05055 (2019).
14. Md Zahangir A, Taha TM, Yakopcic C, Westberg S, Sidike P, Mst Nasrin S, Hasan M, Van Essen BC, Awwal AAS, Asari VK. A state-of-the-art survey on deep learning theory and architectures. *Electronics*. 2019; 8, no. 3: 292.
15. Khan A, Sohail A, Zahoora U, Qureshi AS. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*. 2019: 1-62.

Streszczenie

Ostatnie dziesięciolecie (2010-2019) to niepowstrzymywalny rozwój technik przetwarzania obrazów związanych z sieciami neuronowymi. Powszechne użycie internetu oraz fotografii cyfrowej dostarczyło ogromnej ilości danych do przetworzenia. Szybki rozwój sprzętu oferującego dużą moc obliczeniową (np. procesory GPU) umożliwił za to znaczne obniżenie kosztów przetwarzania danych. Oba te fakty sprawiły, że możliwe stało się szybkie i efektywne trenowanie sieci neuronowych w oparciu o wiedzę zgromadzoną w zbiorach danych obrazowych. Celem opracowania jest przedstawienie wybranych, najnowszych zagadnień z dziedziny przetwarzania obrazów

na poziomie koncepcji, bez przedstawiania dokładnego aparatu matematycznego, i rozważenie w jaki sposób koncepcje te można interpolować na inne obszary przetwarzania obrazów czy też wykorzystać do tworzenia kolejnych idei.

Słowa kluczowe: przetwarzanie obrazów, głębokie uczenie maszynowe, architektury sieci neuronowych

Summary
Neural Networks in Computer Vision: a review of selected advancements

The last decade (2010-2019) is the unstoppable development of image processing techniques associated with neural networks. The widespread use of the internet and digital photography has provided a huge amount of data to process. The rapid development of equipment offering high computing power (e.g. GPUs) has enabled a significant reduction in data processing costs. Both of these facts meant that it became possible to train neural networks quickly and efficiently based on knowledge accumulated in image data sets. The aim of the paper is to present selected, the latest issues in the field of image processing at the concept level, without presenting an exact mathematical apparatus and to consider how these concepts interpolate into other areas of image processing or use to create subsequent ideas.

Keywords: computer vision, deep learning, neural network architectures

Politechnika Białostocka