ARCHIWUM
Akademii Obrony Narodowej

AKADEMIA OBRONY NARODOWEJ

CENTRUM SYMULACJI
I KOMPUTEROWYCH GIER WOJENNYCH

PROSTE NARZĘDZIA INFORMATYCZNE A PROCES PLANOWANIA DZIAŁAŃ

Biblioteka Główna
Akademii Obrony Narodowej
S/5603
05-005603-(

68614

WARSZAWA

AKADEMIA OBRONY NARODOWEJ

CENTRUM SYMULACJI
I KOMPUTEROWYCH GIER WOJENNYCH

BIBLIOTEKA GŁÓWNA - ARCHIWUM
Nr ewid. 68614
Akademii Obrony Narodowej

PROSTE NARZĘDZIA INFORMATYCZNE A PROCES PLANOWANIA DZIAŁAŃ

KRYPTONIM
„PRZEWAGA-1"

opracował:
mjr dr Henryk Spustek

BIBLIOTEKA GŁÓWNA
Nr ewid. ...
Działu Zbiorów Jawnych - Akademii ON

WARSZAWA 2003

SPIS TREŚCI

WPROWADZENIE

W przedstawionym opracowaniu przyjęto następujący schemat postępowania:
Na wstępie podano fizyczną interpretację entropii na prostym przykładzie tasowania kart do gry. Wskazano przy tym na podstawowe problemy opisu zjawisk nieodwracalnych.

Następnie, w rozdziale pierwszym, wskazano na miejsce systemów informatycznych w procesie dowodzenia. Pokazano również zastosowanie prostych narzędzi informatycznych do rozwiązywania wybranych problemów procesu dowodzenia. Bazując na analizie literatury traktującej o problemach pomiaru i wartościowania informacji, omówiono założenia probabilistycznego modelu przewagi informacyjnej. Jednocześnie, zaproponowano użycie symulacyjnej metody Monte Carlo na etapie generacji wartości poszczególnych stanów w macierzy wariantów działania stron. Stanowi to treść rozdziału drugiego.

Rozdział trzeci zawiera opis zaawansowanych mechanizmów obliczeniowych popularnego arkusza kalkulacyjnego Excel, w kontekście ich użycia do współpracy ze środowiskiem programistycznym DELPHI. Treści te pomagają zrozumieć istotę zastosowanych mechanizmów implementacyjnych algorytmu metody określania przewagi w walce i operacji.

W rozdziale czwartym omówiono założenia oraz szczegóły budowy aplikacji wspomagającej proces oceny przewagi w walce i operacji, bazujący na algorytmie opisanym w rozdziale drugim.

Procesy nieodwracalne

Przykłady procesów nieodwracalnych znajdujemy między innymi na kartach książek traktujących o zjawiskach fizycznych. Przykładowo: kawałek kredy spadając ze stołu na podłogę podlegając prawom mechaniki klasycznej przechodzi do innego stanu z którego powrót do stanu pierwotnego, chociaż teoretycznie możliwy, nigdy nie zachodzi. Kawałek kredy mógłby wznieść się z podłogi do góry i osiąść z powrotem na stole. Proces ten nigdy jednak nie zachodzi, a wiemy to z codziennego doświadczenia i z prawa wzrostu entropii.

O entropii i prawie wzrostu entropii

Przykład

*Rozpatrzmy talię kart ułożonych rosnąco od dwójki trefl do asa pik. Talia znajduje się w stanie, który oznaczymy umownie przez **A**. Każdy inny stan kart, po dowolnym ich przetasowaniu, oznaczymy przez **B**. Przejście od stanu A do stanu B jest procesem nieodwracalnym. Wiemy, że przejście to jest teoretycznie możliwe, lecz bardzo mało prawdopodobne.*[1]

Stany mikro i makroskopowe

Powracamy do poprzedniego przykładu z talią kart. Stany **A** i **B** nazywamy makroskopowymi. Stan mikroskopowy stanowi konkretne ułożenie kart.

Przez $\Delta\Gamma$ oznaczymy liczbę stanów mikroskopowych.

Obliczymy ile stanów mikroskopowych odpowiada stanom makroskopowym **A** i **B**.

Stanowi **A** odpowiada tylko jeden stan, więc: $\Delta\Gamma(A)=1$.

Zaś dla stanu **B** mamy: $\Delta\Gamma(B)=52!-1\approx 8\cdot 10^{67}$.

Wniosek

Zakładając, że tasowanie kart jest idealne, tzn. że każdy stan mikroskopowy jest wytwarzany z równym prawdopodobieństwem, startując z jakiegokolwiek stanu, mamy $8\cdot 10^{67}$ razy większe prawdopodobieństwo przejścia do stanu **B** niż do stanu **A**.

Z punktu widzenia praw tasowania możliwe są przejścia zarówno ze stan **A** do **B** jak i z **B** do **A**. Praktycznie jednak, tylko przejścia z **A** do **B** są realizowane. Jeśli układ raz znajdzie się w stanie **B**, to już go nie opuści.

Podane wyżej rozumowanie stosuje się bez zmian jakościowych do wyjaśnienia procesów nieodwracalnych w praktycznie spotykanych układach makroskopowych.

Wielkość $\Delta\Gamma$ gwałtownie rośnie wraz ze wzrostem liczby składników układu. Przykładowo, gdyby do talii kart dodać jedną kartę, to liczba $\Delta\Gamma(B)$ wzrosłaby 53 – krotnie.

[1] Przykład zaczerpnięto z K.Zalewski, Wykłady z termodynamiki fenomenologicznej i statystycznej, PWN Warszawa 1978, s.50.

Stąd, wygodniej jest zamiast liczby $\Delta\Gamma$ wprowadzić entropię S zdefiniowaną jako:

$$S = k \cdot \ln \Delta\Gamma ,$$

gdzie k jest stałą.

Powyższy wzór, zwany wzorem Boltzmanna, stosuje się do **układów izolowanych w równowadze pełnej bądź niepełnej**.

W praktyce, **każdy układ można interpretować jako część jakiegoś większego układu izolowanego**. Dzięki temu, teoria entropii pozwala na wyjaśnienie natury wszystkich procesów nieodwracalnych.[2]

Wracamy do przykładu z uporządkowaną talią kart. Obliczymy entropię stanów **A** i **B**.

$$S(A) = 0 , \qquad S(B) = k \cdot \ln\left(8 \cdot 10^{67}\right) = 156 \cdot k .$$

Zatem, wyjaśnienie procesów nieodwracalnych sprowadza się do twierdzenia, że **w procesach nieodwracalnych, zachodzących w układzie izolowanym, entropia rośnie**. Jest to jedna z postaci prawa wzrostu entropii.

Stan układu izolowanego, w którym układ posiada maksymalną entropię jaką można osiągnąć w danych warunkach, nazywany jest **stanem równowagi pełnej**.

Porównanie entropii układu w stanie pełnej równowagi z entropią układu w innych stanach jest wykonalne tylko wtedy, gdy te inne stany są **stanami równowagi niepełnej**.

Prawdopodobieństwo makrostanu

Dany stan makroskopowy realizuje się za pośrednictwem dużej ilości mikrostanów. Znając cechy charakteryzujące dany makrostan, można obliczyć wszystkie mikrostany, zgodnie z tymi cechami.[3]

Prawdopodobieństwo wystąpienia makrostanu definiuje się poprzez stosunek liczby mikrostanów odpowiadających danemu makrostanowi, do całkowitej liczby stanów osiąganych przez dany układ. Przykładowo dla makrostanu **A** mamy:

$$p_A = \frac{\Delta\Gamma(A)}{\Delta\Gamma} .$$

W naszym przykładzie z talią kart: $p_A = \frac{1}{52!} \approx \frac{1}{8 \cdot 10^{67}} \to 0$, $p_B \approx 1$.

Liczbę mikrostanów $\Delta\Gamma(A)$ nazywa się także prawdopodobieństwem stanu makroskopowego. Liczba ta nie jest prawdopodobieństwem w sensie matematycznym, gdyż to ostatnie musi być mniejsze lub równe jedności, a liczba $\Delta\Gamma(A)$ jest bardzo duża.

---

[2] tamże s.52.

Nazwa „prawdopodobieństwo stanu" wzięła się stąd, że wzór: $p_A = \frac{\Delta\Gamma(A)}{\Delta\Gamma}$ pozwala znajdować prawdopodobieństwo makrostanu.[4]

Wychodząc z rozważań kombinatorycznych, wykorzystując wzór Stirlinga[5] możliwym staje się obliczenie prawdopodobieństwa makrostanu.

---

[3] A.M.Matwiejew, Fizyka cząsteczkowa, PWN, Warszawa 1989, s.49.

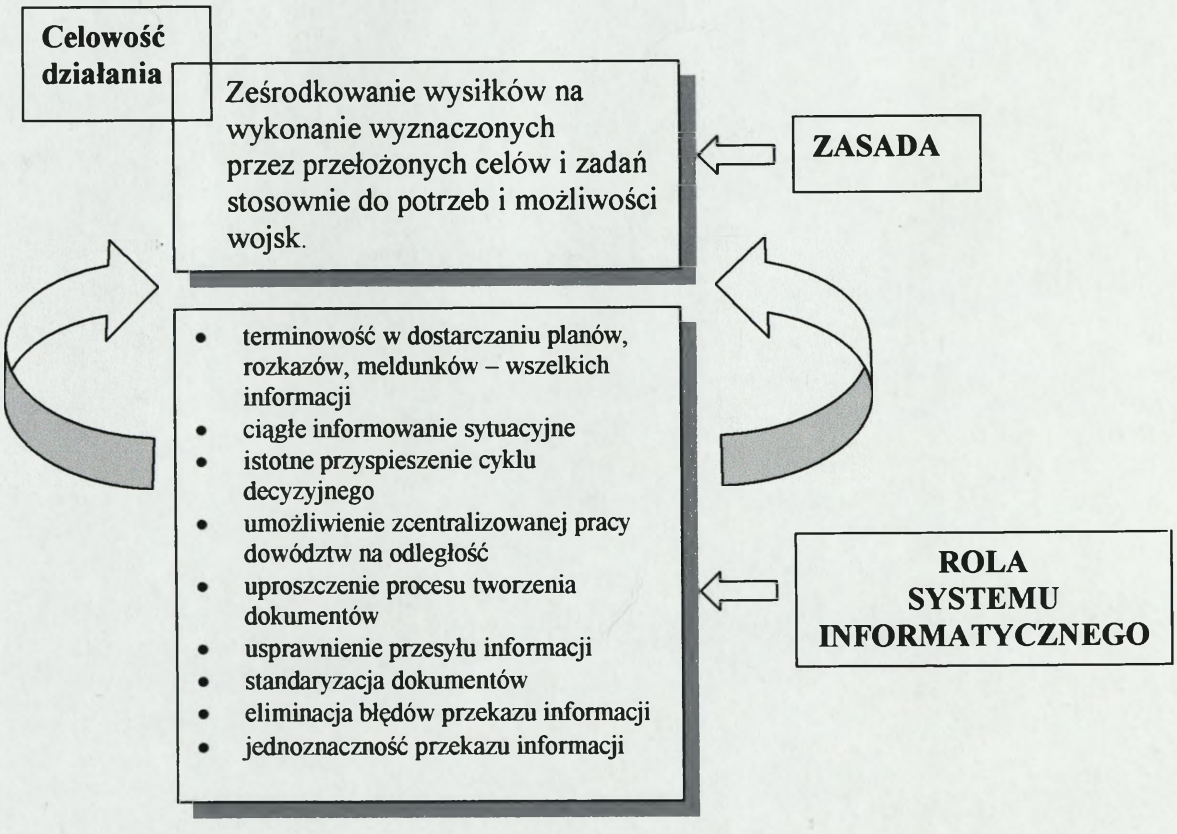
[4] Bezpośrednie policzenie stanów przeważnie jest niemożliwe. Stąd, teoria znajdowania ilości stanów powinna być tak skonstruowana by uniknąć konieczności ich liczenia. Najkorzystniej byłoby znaleźć od razu prawdopodobieństwa p_A bez znajomości liczby stanów $\Delta\Gamma(A)$. Metodami bezpośredniego obliczania liczby mikrostanów są metody kombinatoryczne. Wykorzystujemy zależności na liczbę możliwych kombinacji lub wariacji, zależnie od natury rozpatrywanego zjawiska:

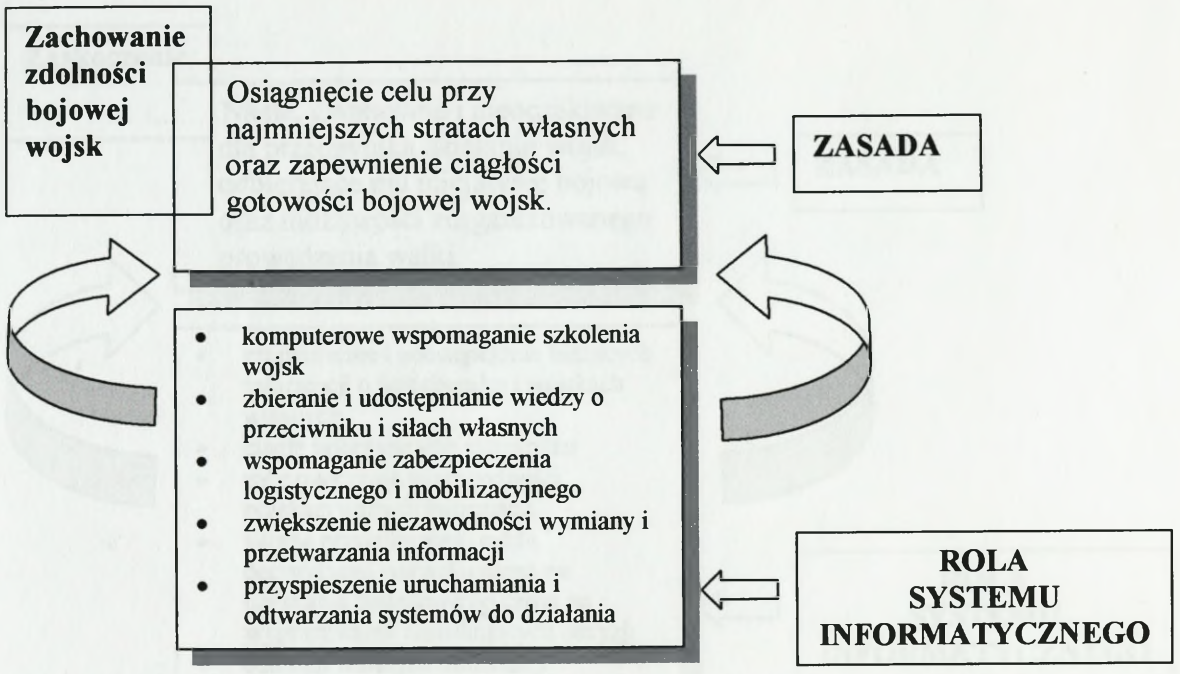
$$C(n,m) = \binom{n}{m} = \frac{n!}{m!(n-m)!}, \quad V(n,m) = \frac{n!}{(n-m)!}, \quad \overline{V(n,m)} = n^m.$$

[5] Dla dużych n spełniona jest równości: $n! \approx (n/e)^n$, $\ln n! \approx \ln\left[(n/e)^n\right] = n\ln(n/e) = n\ln n - n$.

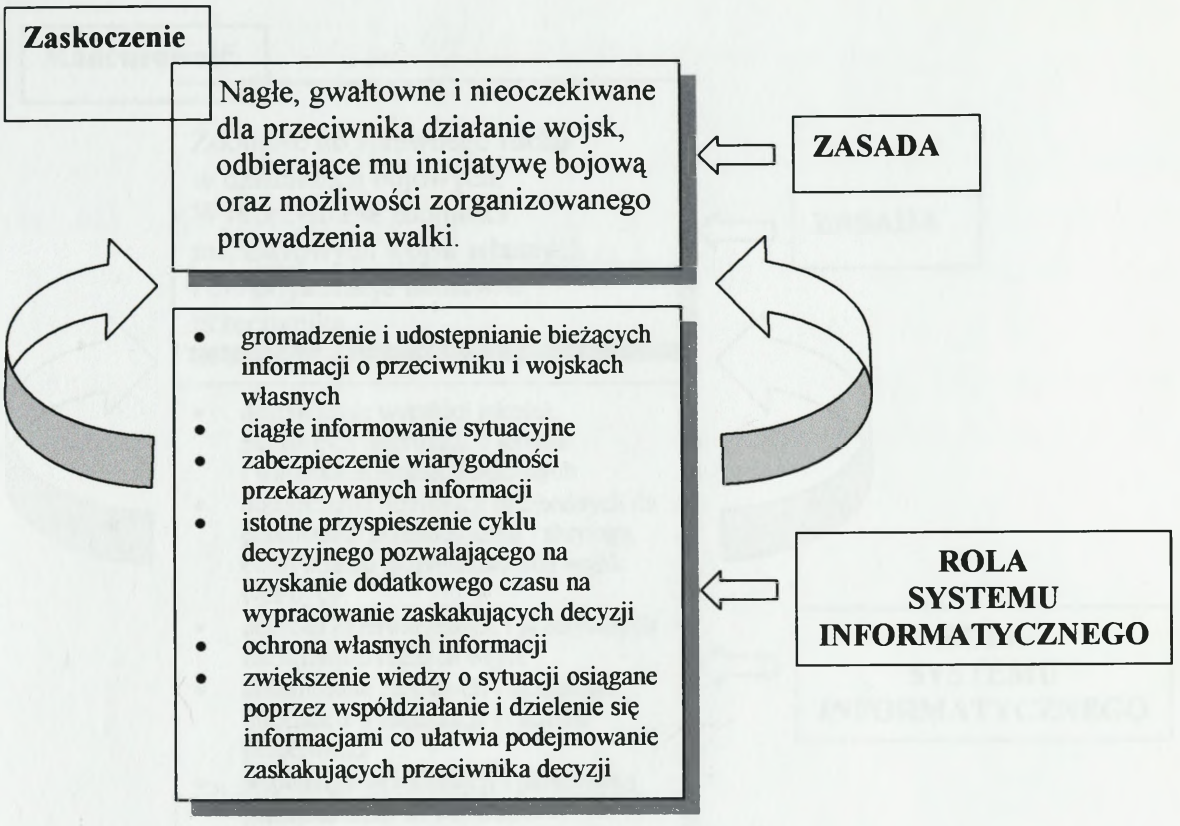
1. NARZĘDZIA INFORMATYCZNE A PROBLEMY WSPÓŁCZESNEGO POLA WALKI

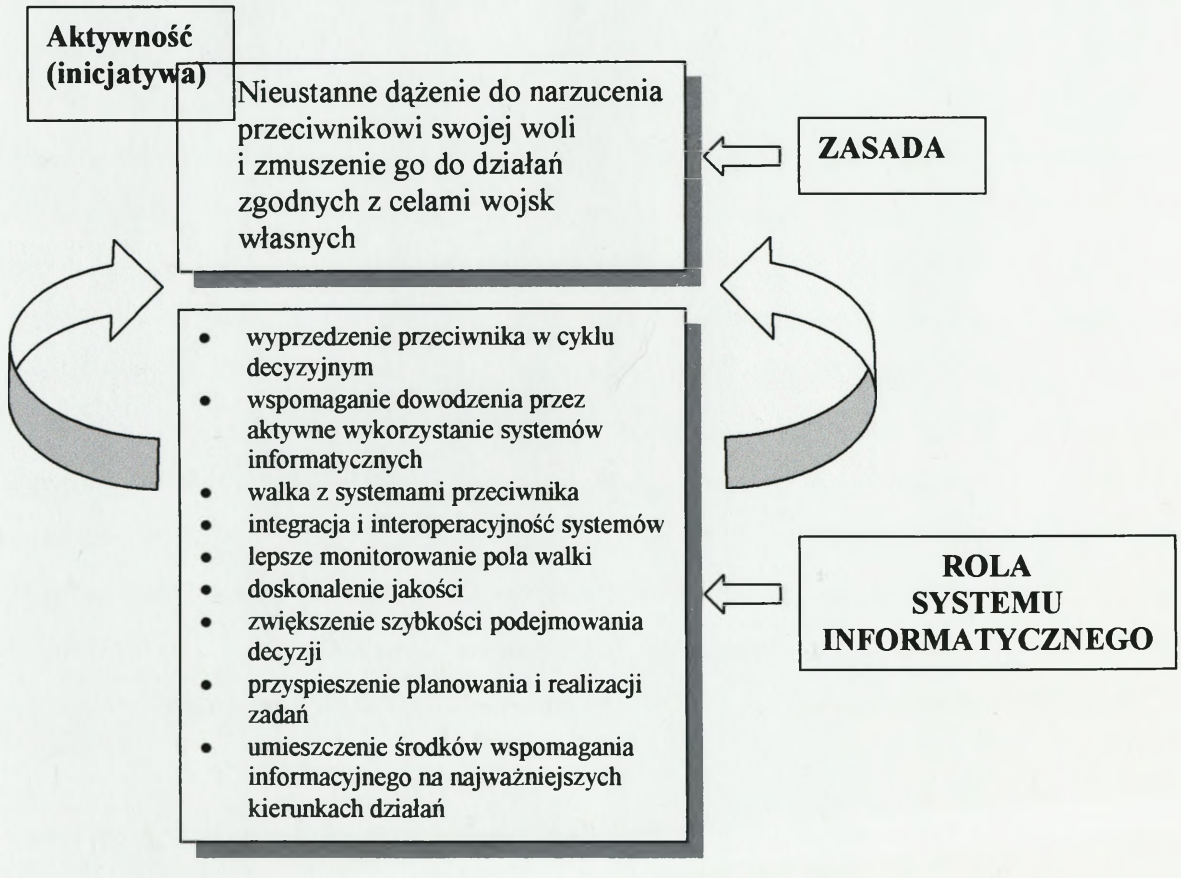
Regulamin działań taktycznych wojsk lądowych precyzuje warunki uzyskania i utrzymania przewagi ilościowej i jakościowej nad przeciwnikiem. Znajdujemy tam, że warunkiem uzyskania, utrzymania i wykorzystania przewagi ilościowej i jakościowej nad przeciwnikiem tak, aby z jak najmniejszymi stratami osiągnąć cel walki, operacji i wojny w możliwie najkrótszym czasie, jest stosowanie zasad sztuki wojennej. Stosowanie zasad sztuki wojennej ma znaczenie nie tylko podczas działań wojennych, ale także podczas pokojowych operacji wojskowych. Narzędzia wsparcia informatycznego powinny umożliwić zabezpieczenie realizacji zasad sztuki wojennej.

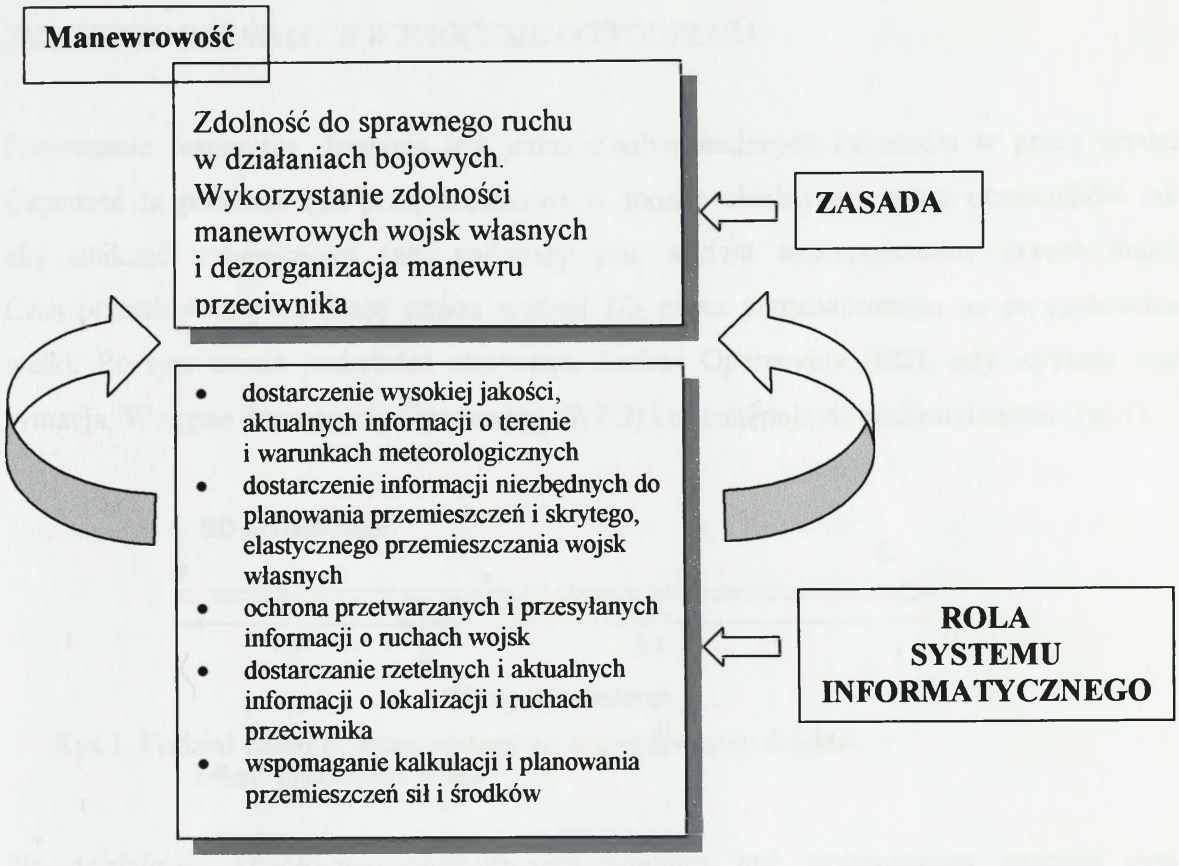



Zachowanie zdolności bojowej wojsk
Osiągnięcie celu przy najmniejszych stratach własnych oraz zapewnienie ciągłości gotowości bojowej wojsk.
ZASADA
komputerowe wspomaganie szkolenia wojsk
zbieranie i udostępnianie wiedzy o przeciwniku i siłach własnych
wspomaganie zabezpieczenia logistycznego i mobilizacyjnego
zwiększenie niezawodności wymiany i przetwarzania informacji
przyspieszenie uruchamiania i odtwarzania systemów do działania
ROLA SYSTEMU INFORMATYCZNEGO


Ekonomia sił
Racjonalne dysponowanie siłami i środkami odpowiednio do ważności zadań, tak by osiągnąć cel w możliwie najkrótszym czasie, przy najmniejszych stratach własnych.
ZASADA
1 Zwiększenie elastyczności użycia sił i środków poprzez:
wspomaganie zabezpieczenia logistycznego,
przyspieszenie cyklu decyzyjnego.
2. Uzyskanie przewagi informacyjnej
3. Udostępnienie wiedzy o przeciwniku i siłach własnych.
4. Opracowanie wariantów działania i wyboru rozwiązań optymalnych.
5. Zwiększenie szybkości działania co pozwoli na skupienie sił w odpowiednim miejscu i czasie.
6. Zwiększenie dostępności aktualnych informacji.
7. Zwiększenie efektywności niszczenia.
ROLA SYSTEMU INFORMATYCZNEGO


Zaskoczenie
Nagłe, gwałtowne i nieoczekiwane dla przeciwnika działanie wojsk, odbierające mu inicjatywę bojową oraz możliwości zorganizowanego prowadzenia walki.
ZASADA
• gromadzenie i udostępnianie bieżących informacji o przeciwniku i wojskach własnych
• ciągłe informowanie sytuacyjne
• zabezpieczenie wiarygodności przekazywanych informacji
• istotne przyspieszenie cyklu decyzyjnego pozwalającego na uzyskanie dodatkowego czasu na wypracowanie zaskakujących decyzji
• ochrona własnych informacji
• zwiększenie wiedzy o sytuacji osiągane poprzez współdziałanie i dzielenie się informacjami co ułatwia podejmowanie zaskakujących przeciwnika decyzji
ROLA SYSTEMU INFORMATYCZNEGO

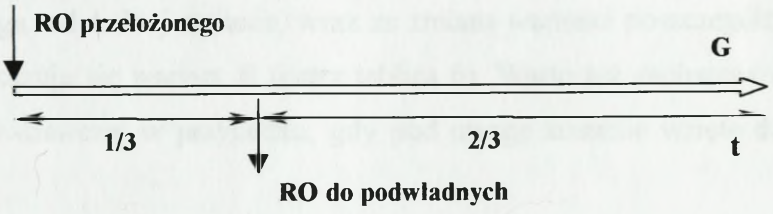

Aktywność (inicjatywa)
Nieustanne dążenie do narzucenia przeciwnikowi swojej woli i zmuszenie go do działań zgodnych z celami wojsk własnych
ZASADA
• wyprzedzenie przeciwnika w cyklu decyzyjnym
• wspomaganie dowodzenia przez aktywne wykorzystanie systemów informatycznych
• walka z systemami przeciwnika
• integracja i interoperacyjność systemów
• lepsze monitorowanie pola walki
• doskonalenie jakości
• zwiększenie szybkości podejmowania decyzji
• przyspieszenie planowania i realizacji zadań
• umieszczenie środków wspomagania informacyjnego na najważniejszych kierunkach działań
ROLA SYSTEMU INFORMATYCZNEGO


Manewrowość
Zdolność do sprawnego ruchu w działaniach bojowych. Wykorzystanie zdolności manewrowych wojsk własnych i dezorganizacja manewru przeciwnika
ZASADA
dostarczenie wysokiej jakości, aktualnych informacji o terenie i warunkach meteorologicznych
dostarczenie informacji niezbędnych do planowania przemieszczeń i skrytego, elastycznego przemieszczania wojsk własnych
ochrona przetwarzanych i przesyłanych informacji o ruchach wojsk
dostarczanie rzetelnych i aktualnych informacji o lokalizacji i ruchach przeciwnika
wspomaganie kalkulacji i planowania przemieszczeń sił i środków
ROLA SYSTEMU INFORMATYCZNEGO

Przykład

ZNACZENIE INFORMACJI W PROCESIE DOWODZENIA.

Porównanie wariantów działania jest jedną z najważniejszych czynności w pracy sztabu. Czynność ta powinna być przeprowadzona w sposób obiektywny przez uczestników tak, aby uniknąć stronniczości (np. najlepszy jest wariant zaproponowany przeze mnie). Czas przewidywany na pracę sztabu wynosi 1/3 czasu przeznaczonego na przygotowanie walki. Po tym czasie podwładni otrzymują Rozkaz Operacyjny (RO), gdy wymaga tego sytuacja, Wstępne Zarządzenie Operacyjne (WZO) i przystępują do realizacji zadań (rys.1).



Rys.1. Podział czasu przeznaczonego na przygotowanie działań.
Źródło: opracowanie własne.

Na działalność koncepcyjno-organizacyjną powinna być przeznaczona znaczna część dostępnego czasu, a tylko nieznaczna jego część, na przedsięwzięcia kalkulacyjne czy redakcyjne.

Pełna kontrola dowództw i sztabów nad potokiem informacji, zapewnienie ich szybkiego przetworzenia, dokonanie niezbędnych prognoz oraz sporządzenie różnorodnych dokumentów, przy jednoczesnym zapewnieniu wysokiego stopnia ich wiarygodności, bez wspomagania informatycznego stało się wręcz niemożliwe. Dynamicznemu rozwojowi techniki towarzyszą rosnące wymagania osób wykorzystujących jej osiągnięcia. Żądane jest zapewnienie możliwości przekazywania danych w sposób wiarygodny, precyzyjny i aktualny. Uczestnicy tego procesu chcą mieć pewność co do poprawności przetwarzanych danych i rzetelności otrzymanych wyników. Stosując narzędzia informatyczne oczekują oni lepszych efektów swojej pracy w ograniczeniach czasowych, czy wręcz przy braku znajomości złożonych metod matematycznych ograniczając się jedynie do obsługi programu.

Warunkiem informatycznego wspomożenia omawianych zagadnień jest zbudowanie ścisłego w sensie matematycznym algorytmu opartego na wybranej metodzie analitycznej.[6]

---

[6] prezentowana tutaj koncepcja wparcia procesu podejmowania decyzji została zaprezentowana przez autora na 13-tej międzynarodowej konferencji naukowej ITEC 2002 w Lille (Francja).

Do analiz wybrano metodę taksonomii numerycznej opartej na algorytmie opracowanym przez Hugo Steinhausa. Wykonano stosowne obliczenia wykorzystując dane zaczerpnięte z literatury.[7]

Jednym z istotnych kryteriów w przeprowadzonej analizie porównawczej jest informacja a raczej stopień jej posiadania. Stąd kryterium to mieści się w kategorii stymulant.

Wyniki obliczeń przedstawiono poniżej. Informacyjnie podano również wyniki pośrednie. W rzeczywistości, użytkownik wprowadza jedynie dane wejściowe i otrzymuje końcowy wynik analizy. Pozostałe, wymienione kryteria mieszczą się w kategorii stymulant.

Ocena wariantów B i C okazała się w tym przypadku zbliżona (z lekką przewagą na korzyść wariantu C). Powodem takiego stanu rzeczy jest narzucony w przykładzie układ wag.

Sytuacja ulega radykalnej zmianie, wraz ze zmianą wartości poszczególnych wag. Wówczas to, lepszy okazuje się wariant B (patrz tablica 6). Warto też zaobserwować zmiany wyniku analizy porównawczej w przypadku, gdy pod uwagę zostanie wzięte dodatkowe kryterium (tablica 7).

Przy okazji warto zauważyć, że wprowadzone dodatkowo kryterium mieści się w kategorii destymulant, co w żadnym wypadku nie utrudnia analizy.

Inny problem, na który warto zwrócić uwagę związany jest z systemem nadawania wag, tzn. chodzi tu o subiektywizm w ocenach wartości poszczególnych wag.[8]

[7] J.Michniak, Metody i treści pracy zespołów funkcjonalnych na SD WLąd, AON Warszawa 2000, s.68.
[8] E. Gatnar, Symboliczne metody klasyfikacji danych PWN, Warszawa 1998, s.35.

Tablica 1. Rozważane warianty działania.

Kryterium	Znaczenie kryterium	Wariant „A”	Wariant „B”	Wariant „C”
Prostota	**2**	2 / **4**	1 / **2**	3 / **6**
Zaskoczenie	**3**	1 / **3**	3 / **9**	2 / **6**
Czas	**5**	1 / **5**	2 / **10**	3 / **15**
Ekonomia sił	**1**	1 / **1**	2 / **2**	1 / **1**
Wsparcie logistyczne	**2**	1 / **2**	3 / **6**	2 / **4**
Działania połączone	**1**	1 / **1**	2 / **2**	1 / **1**
Suma/Suma po uwzględnieniu kryterium		1 / **16**	13 / **31**	12 / **33**

Źródło: Praca zespołowa pod kier. J. Michniak, Metody i treść pracy zespołów funkcjonalnych na SD WLąd , AON, Warszawa 2000, s.68.

Tablica 2. Dane do porównania wariantów działania.

	Prostota	Zaskoczenie	Czas	Ekonomia sił	Wsparcie logistyczne	informacja
Wariant „A”	**4**	**3**	**5**	**1**	**2**	**1**
Wariant „B”	**2**	**9**	**10**	**2**	**6**	**2**
Wariant „C”	**6**	**6**	**15**	**1**	**4**	**1**
Rodzaj cechy	**S**	**S**	**S**	**S**	**S**	**S**
Wagi	**0,15**	**0,2**	**0,36**	**0,07**	**0,15**	**0,07**

Źródło: opracowanie własne.

Tablica 3. Macierz ustandaryzowanych cech

	Prostota	Zaskoczenie	Czas	Ekonomia sił	Wsparcie logistyczne	Informacja
Wariant „A”	**0,000**	**-1,225**	**-1,225**	**-0,707**	**-1,225**	**-0,707**
Wariant „B”	**-1,225**	**1,225**	**0,000**	**1,414**	**1,225**	**1,414**
Wariant „C”	**1,225**	**0,000**	**1,225**	**-0,707**	**0,000**	**-0,707**

Źródło: opracowanie własne.

Tablica 4. Macierz dyspersji.

	Prostota	Zaskoczenie	Czas	Ekonomia sił	Wsparcie logistyczne	Informacja
Wariant „A”	**1,500**	**-6,000**	**6,000**	**4,500**	**6,000**	**4,500**
Wariant „B”	**6,000**	**0,000**	**1,500**	**0,000**	**0,000**	**0,000**
Wariant „C”	**0,000**	**1,500**	**0,000**	**4,500**	**1,500**	**4,500**

Źródło: opracowanie własne.

Tablica 5. Wektor średniej odległości (d_{on}) z wynikami oceny globalnej.

	Wariant „A”	Wariant „B”	Wariant „C”
d_{on}	**2,262**	**1,200**	**1,075**
Wynik globalny	**0,273**	**0,614**	**0,654**

Źródło: opracowanie własne.

Tablica 6. Tabela danych do porównania wariantów działania. Przykład drugi, uwzględniający zmianę wag.

	Prostota	Zaskoczenie	Czas	Ekonomia sił	Wsparcie logistyczne	Informacja	Wynik globalny
Wariant „A”	4	3	5	1	2	1	0,320
Wariant „B”	2	9	10	2	6	2	0,729
Wariant „C”	6	6	15	1	4	1	0,472
Rodzaj cechy	S	S	S	S	S	S	
Wagi	0,1	0,1	0,1	0,3	0,15	0,25	

Źródło: opracowanie własne.

Tablica 7. Tabela danych do porównania wariantów działania. Przykład trzeci, uwzględniający zmianę wag i dodatkowe kryterium.

	Prostota	Zaskoczenie	Czas	Ekonomia sił	Wsparcie logistyczne	Informacja	Oddziaływa -nie p-ka	Wynik globalny
Wariant „A”	4	3	5	1	2	1	6	0,392
Wariant „B”	2	9	10	2	6	2	6	0,404
Wariant „C”	6	6	15	1	4	1	4	0,766
Rodzaj cechy	S	S	S	S	S	S	D	
Wagi	0,2	0	0	0	0,1	0,1	0,6	

Źródło: opracowanie własne.

2. ALGORYTM ROZWIĄZANIA PROBLEMU OKREŚLANIA PRZEWAGI W WALCE I OPERACJI

Model prawdopodobieństwa wiedzy

Analizując dostępną literaturę, zwrócono uwagę na opracowanie poświęcone teoretycznym podstawom przewagi informacyjnej. Stworzono tam pewien model prawdopodobieństwa wiedzy który posłużył jako podstawa rozważań teoretycznych. Następnie odwołano się do teorii gier i równań Lanchester'a. Do dalszych rozważań wykorzystano model prawdopodobieństwa wiedzy wraz ze zdefiniowanymi miarami informacji, celem stworzenia zależności analitycznych opisujących przewagę informacyjną nad przeciwnikiem. Zdefiniowano pewną nową Miarę Efektywności (MOE) opartą na wiedzy, jaką jest kontrola przestrzeni bojowej dla koncepcji manewru dominującego. Zbadano także możliwość opracowania nowych Miar Efektywności dla poszczególnych rodzajów operacji.[9]

Prace nad miarami efektywności i związanymi z nimi wielkościami wymiernymi zdecydowanie sugerują, że informacja - w szczególności przewaga informacyjna, może mieć silny wpływ na wyniki operacji militarnych. Pomiar stopnia przewagi informacyjnej, jaki mogłaby osiągnąć, jedna strona nad drugą, jest czymś najbardziej pożądanym w wieku informacyjnym. W raporcie, o którym tu mowa, skupiono się na miarach relatywnych, poczynając od wiedzy względnej, dla której opracowano jednostkę miary wiedzy.
Ta wymierna wielkość, wyraża zależność między wiedzą idealną i rzeczywistą w operacjach militarnych, dla obu stron. Skupiono się także na potrzebie posiadania nowych miar efektywności do oceny nowych koncepcji operacji aktualnie przyjmowanych przez Armię, jak również na wpływie, jaki na te operacje może wywierać informacja.
Tradycyjne miary efektywności (wciąż „zakorzenione" w modelach opracowywanych dla każdego indywidualnego układu sił), wyliczają efektywność na bazie wskaźników zmian zdominowanych przez główne systemy uzbrojenia. Mierzą one jedynie pewną część zdolności bojowej dowolnej jednostki wojskowej. Ponadto, tradycyjne miary efektywności nie spełniają

[9] R. Darilek i inni, Miary efektywności dla armii wieku informacyjnego, Centrum Rand Arroyo, 2002 – tłumaczenie z języka angielskiego, źrdło: http//www.rand.org/organization/ard/

swojego zadania, gdy dochodzi do przygotowywania operacji stabilizujących i zachowania bezpieczeństwa - operacji znanych wcześniej jako operacje militarne inne niż wojna (MOOTW), które mogą zdominować przyszłe operacje militarne.

Dwa warianty prawa Lanchestera (wariant liniowy i kwadratowy), zostały uzupełnione przez tzw. wariant mieszany - prawo Lanchestera zmodyfikowane wiedzą.

W omawianym tu podejściu probabilistycznym, widoczny staje się pewien mechanizm zwany „fizykalizacją zjawisk".[10] Mechanizm ten jest naturalną konsekwencją stosowania aparatu matematycznego do opisu różnych zjawisk analogicznie do tych, występujących w fizyce klasycznej. Tego typu podejście wynika z chęci zastosowania „eleganckich" w sensie analitycznym metod analiz zjawisk z jakimi mamy do czynienia w fizyce. Stąd zastosowanie „hamiltonianów" i koncepcji entropowego modelu informacji w połączeniu z prawem Lanchestera zmodyfikowanym wiedzą.

Kolejne modyfikacje równań Lanchestera dowodzą, ograniczonej ich przydatności do opisu rzeczywistych zjawisk pola walki. Nie można odmówić tym równaniom „elegancji" w zapisie, jednakże pozostaną one jedynie narzędziem dociekań akademickich. Prawdziwych rozwiązań należy poszukiwać w symulacjach komputerowych.

Metoda Monte Carlo w probabilistycznym modelu przewagi

Metoda Monte Carlo polega na przedstawieniu rozwiązania podstawowego problemu w postaci parametru pewnej hipotetycznej populacji i używaniu losowej sekwencji liczb do tworzenia próbki tej populacji, na podstawie której można dokonać statystycznego oszacowania wartości badanego parametru.[11]

Algorytm metody Monte Carlo[12]

Dwanaście kroków postępowania w metodzie Monte Carlo:

- Określamy parametr stanowiący podstawowy miernik danego, rozwiązywanego problemu.
- Budujemy model analityczny danego problemu, wykorzystując matematyczne zależności pomiędzy najważniejszymi zmiennymi. Zmienne w modelu mogą mieć

[10] patrz H. Spustek, Przewaga w walce i operacji, rozprawa habilitacyjna AON, Warszawa 2002 r.
[11] Dieter W.Heermann, Podstawy symulacji komputerowych w fizyce, WNT Warszawa 1997 s.80.
[12] F.Fabozzi, Fixed income mathematics, Probus Publishing Co., 1997.

charakter deterministyczny lub losowy. Zmienne deterministyczne mogą przyjmować tylko jedną wartość, podczas gdy zmienne losowe mogą przybierać wiele wartości.

- Dla każdej zmiennej losowej musi być określony odpowiadający jej rozkład prawdopodobieństwa.
- Rozkład prawdopodobieństwa każdej zmiennej losowej musi być przetworzony do postaci skumulowanego rozkładu prawdopodobieństwa.
- Każdej możliwej wartości zmiennej losowej musi być przypisana odpowiednia wartość losowa, ze skumulowanego rozkładu prawdopodobieństwa tej zmiennej.
- Dla każdej zmiennej losowej musi istnieć możliwość wygenerowania liczby losowej.
- Każdej liczbie losowej musi być przypisana odpowiednia wartość zmiennej losowej.
- Odpowiednia wartość zmiennej losowej, określona w poprzednim kroku, musi być wykorzystana do wyznaczania podstawowego miernika danego problemu zgodnie z krokiem drugim niniejszego algorytmu.
- Wartość wyznaczona dla podstawowego miernika w kroku ósmym (poprzednim) musi zostać zapamiętana.
- Kroki od sześć do dziewięć muszą być powtarzane wymaganą ilość razy (zazwyczaj od 100 do 1000 razy).
- Wartości podstawowego miernika, zapamiętane zgodnie z punktem dziewiątym, stają się podstawą do określania jego rozkładu prawdopodobieństwa i skumulowanego rozkładu prawdopodobieństwa.
- Skumulowany rozkład prawdopodobieństwa utworzony w punkcie jedenastym musi zostać przeanalizowany. Podczas analizy wyznaczane są wybrane parametry statystyki opisowej.

Metoda Monte Carlo w zastosowaniu do generacji wartości stanów w macierzy wariantów działania stron

„Zmodyfikujemy" dwanaście kroków postępowania do poniższych sześciu:

- Określ punkt początkowy x_0 w przestrzeni fazowej.
- Wygeneruj nowy stan x'.
- Oblicz prawdopodobieństwo przejścia $W(x, x')$.
- Wygeneruj liczbę losową $R \in [0,1]$ (rozkład równomierny).
- Jeżeli prawdopodobieństwo przejścia W jest mniejsze niż liczba losowa R, to potraktuj stary stan jako nowy i wróć do kroku drugiego.
- W przeciwnym razie, zaakceptuj nowy stan i wróć do kroku drugiego.

W metodzie Monte Carlo (wariant opisany powyżej) milcząco założono, że przyjęty wymiar prawdopodobieństw przejść spełnia **wymóg ergodyczności**. Termin **ergodyczność** oznacza, że każdy stan jest osiągalny z każdego innego stanu. Ściślej mówiąc, każdy stan musi być dostępny z każdego innego stanu w skończonej liczbie przejść. W innych przypadkach, stany układu są podzielone na różne klasy ergodyczności i nie jest możliwe przejście między tymi klasami.[13]

Przykład

Dana jest macierz stochastyczna:

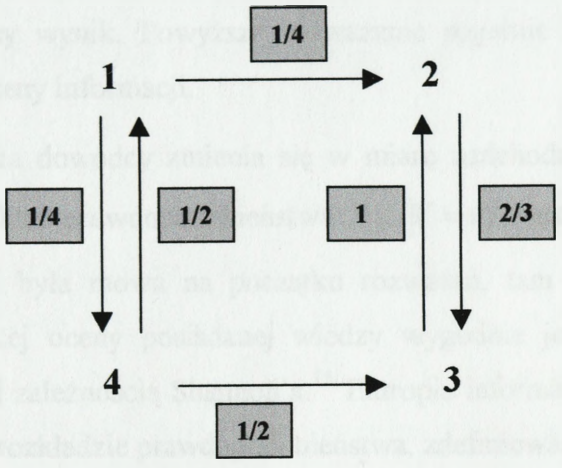
$$\begin{bmatrix} 1/2 & 1/4 & 0 & 1/4 \\ 0 & 1/3 & 2/3 & 0 \\ 0 & 1 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \end{bmatrix},$$

zawierająca prawdopodobieństwa przejścia pomiędzy czterema stanami: 1, 2, 3 i 4.[14]

Przejście ze stanu nr 1 do stanu nr 4 jest możliwe z prawdopodobieństwem $1/4$, *podczas gdy przejście ze stanu nr 4 do stanu nr 1 dokonuje się z prawdopodobieństwem* $1/2$. *Macierz powyższa jest nieredukowalna. Nie ma tu możliwości przejścia ze stanu nr 2 do stanu nr 1 lub stanu nr 4. Ponadto, jeżeli stanami początkowymi są stany nr 2 lub nr 3, to wszystkimi następnymi stanami są albo stan nr 2 albo stan nr 3. Sytuację tą przedstawia*

[13] Dieter W.Heermann, Podstawy symulacji komputerowych w fizyce, WNT Warszawa 1997 s.88.
[14] tamze s.71.

diagram na rysunku niżej. Na diagramie zaznaczono jedynie prawdopodobieństwa przejść z poszczególnych stanów, pominięto prawdopodobieństwa pozostawania w danych stanach.



Rys.2. Diagram prawdopodobieństw przejść pomiędzy stanami 1, 2, 3 i 4.
Źródło: Dieter W.Heermann, Podstawy symulacji komputerowych w fizyce, WNT Warszawa 1997 s.71.

Entropowy model przewagi

Informacja ma dwa istotne atrybuty: wartość i jakość. Informacja przedstawia sobą wartość jeśli informuje dowódcę i dodaje nowe elementy do wiedzy posiadanej przez niego o sytuacji bojowej. Konsekwencją tego jest to, że jeżeli odwołuje się on do "wiedzy" to oznacza to, że jest ona istotna. Jakość informacji zależy od jej precyzji, dokładności, aktualności i kompletności.

W trakcie gromadzenia informacji z różnych źródeł, dowódca poszukuje takich informacji, które posiadają wartość (są one określane terminem CEI – critical elements of information). Problem polega na tym, że rzadko jesteśmy w stanie trafnie ocenić jakość informacji jaką właśnie otrzymujemy. W konsekwencji tego, dowódca musi zdawać sobie sprawę z tego, że część "wiedzy" może być zbyteczna. Przypuśćmy, że przeciwnik używając zaawansowanych technik maskowania i imitacji spowodował, że nasz dowódca zna rzeczywiście położenie połowy obiektów, spośród tych, których sądzi że zna. Powoduje to powstanie kilku dodatkowych kwestii, które musi uwzględnić dowódca w trakcie podejmowania decyzji.

Jeśli podejrzewa on, że jest wprowadzony w błąd, może zdecydować, że należy poczekać do momentu w którym dotrą do niego pewniejsze informacje. Jeśli zaś nic nie podejrzewa, to może postąpić ponownie jak przedtem, otrzymując inny i prawdopodobnie mniej imponujący wynik. Powyższe rozważanie sugeruje konieczność zastosowania użytecznej metody oceny informacji.

Wiedza dowódcy zmienia się w miarę nadchodzących analiz pola walki. Zmienia się zatem rozkład prawdopodobieństwa $P_i(U|V=v)$, możliwych do osiągnięcia „mikrostanów" o których była mowa na początku rozważań, tam gdzie rozważaliśmy entropię układu. Do bieżącej oceny posiadanej wiedzy wygodnie jest użyć wielkość entropię informacji określonej zależnością Shannon'a.[15] Entropia informacji jest miarą średniej ilości informacji w danym rozkładzie prawdopodobieństwa, zdefiniowana następująco:

$$S\left[P_i(U|V=v)\right] = H_i(U|V=v) = -\sum_{u=0}^{n} P_i(U=u|V=v)\ln\left[P_i(U=u|V=v)\right].$$

Funkcja entropii informacji przyjmuje wartość maksymalną, gdy dana informacja przy danym rozkładzie prawdopodobieństwa jest najmniej prawdopodobna (osiąga najwyższy poziom niepewności). Z operacyjnego punktu widzenia, ma to miejsce wówczas, gdy dowódca nie posiada żadnej wiedzy o rozmieszczeniu jakiegokolwiek spośród n celów przeciwnika. W takim przypadku mamy: $P_i(U=u) = \frac{1}{n+1}$. Można łatwo sprawdzić, że entropia w tym rozkładzie wynosi $S_0(U) = \ln(n+1)$. Z drugiej strony, jeśli dowódca posiada potwierdzenie, o rozmieszczeniu μ jednostek w promieniu kontrolowanym, wówczas:

$$P_i(U=\mu|V=v) = 1 \quad i \quad P_i(U\neq\mu|V=v) = 0.$$

Łatwo też można sprawdzić, że entropia w tym rozkładzie jest równa zeru (porównaj rozważania na temat entropii rozkładu kart do gry, prowadzone na początku).

---

[15] Rozumowanie Shannon'a bazuje na entropii zdefiniowanej w termodynamice statystycznej.
Przykład: (M.Cichy, J.Nimańczuk, S.Szpakowicz, Zbiór zadań z propedeutyki informatyki, PWN Warszawa 1977, s.7)
Źródło nadaje n komunikatów z prawdopodobieństwami odpowiednio: $p_1, p_2, p_3, \ldots, p_n$.
Obliczyć ilość informacji zawartą w każdym z komunikatów i entropię źródła.
Ilość informacji zawartej w komunikacie o prawdopodobieństwie p jest określona wzorem:

$$k = \log_2 \frac{1}{p} = -\log_2 p.$$

Entropię informacyjną źródła można obliczyć ze wzoru: $H = \sum_{i=1}^{n} p_i \cdot \log_2 \frac{1}{p_i}$.

Zatem, mając informację o rozmieszczeniu μ jednostek w promieniu kontrolowanym, stopień pewności wynosi: $\ln(n+1) - S_i(U|V=v)$.

Wówczas możemy podać miarę wiedzy w postaci: $S_i(U|V=v) = \frac{n(n+1) - S_i(U|V=v)}{\ln(n+1)}$.

Przykład

*Niech **K** będzie miarą wartości informacji lub wiedzy dostępnej dla dowódcy. W niektórych przypadkach **K** może być wartością liczbową, przykładowo: jeśli przeciwnik posiada N oddziałów, jednostek, wówczas **K=0.3** N. Oznacza to że dowódca zna położenia **0.3N** wrogich oddziałów. Dla obu stron **K** zawiera dwa komponenty : wiedzę o wiedzy wysokiej jakości i wiedzę o wiedzy niskiej jakości lub bezwartościowej wiedzy $K = K_C + K_i$.*

*W przykładzie: $K_C = K_i = 0{,}15$ N. W typowych przypadkach **K** jest wielowymiarowe, zawiera wiele informacji takich jak wolę walki przeciwnika, identyfikację jednostek itp. Warto odnotować że w większości przypadków dowódcy nie wiedzą że są częściowo w błędzie, wyprowadzeni w pole, przez działania przeciwnika.*

W walce, obszar działania (AO- area of operation) , wiedza i manewr współdziałają ze sobą, tak aby zapewnić efektywne wykorzystanie siły ognia. Kiedy jednostka pojawia się w obszarze działań możemy oczekiwać, że będzie zorientowana na działania ofensywne bądź też defensywne. Przeciwnik jest wówczas pełnym graczem, chcącym w aktywny sposób osiągnąć postawione sobie cele. Jednocześnie, stara się on uniemożliwić realizację naszych celów.

KILKA DEFINICJI

Definicja

Oddział kontroluje obszar jeśli jest w stanie operować w nim bez przeszkód. To nie oznacza, że przeciwnik jest wyeliminowany z tego obszaru, a jedynie że zaprzyjaźnione oddziały są w stanie wywierać wpływ na wszystkie punkty w danym obszarze przez cały czas.

Definicja

Promień kontroli oddziału jest to minimum z: maksymalnego efektywnego zasięgu ognia danego oddziału i ognia systemów broni wsparcia w_i, maksymalny efektywny zasięg lokalnych systemów rozpoznania s_i i promienia skojarzonego promienia operacji c_i.

$$r_i = \min\{w_i, s_i, c_i\}$$

Definicja

Wiedza jest to stopień w jakim dowódca jednostki zna położenie jednostek przeciwnika i własnych w swoim promieniu kontroli.

Oznaczmy wiedzę jednostek niebieskich i oraz czerwonych j jako $K_{B,i}$ oraz $K_{R,j}$.

Świadomość sytuacyjna określona w powyższych definicjach może zostać porównana do wiedzy o CEI i może zawierać tak trudne elementy jak ocena postawy bojowej przeciwnika oraz jego intencje. CEI lub istotny element informacji są składnikami informacji koniecznymi do wyznaczenia wspólnego obrazu przestrzeni bojowej i stopnia w jakim ten obraz jest jasny dla dowódcy jednostki, oceniającego swoją świadomość sytuacyjną lub wiedzę.

W naszych rozważaniach, bardziej istotnym jest liczba obcych celi w j – tym celu zainteresowania (TAI – target area of interest), niż wiedza o położeniu n celi w obszarze promienia kontroli.

W przypadku gdy liczba obcych celi w każdym kontrolowanym obszarze jest równa μ_j, wówczas mamy $\sum_j \mu_j \le n$.

Przykład

Dla uproszczenia zakładamy, że TAIs nie nakładają się na siebie, wobec czego suma wiedzy walczących ze sobą dowódców jest w przybliżeniu równa całkowitej liczbie celów przeciwnika w TAIs. Dla n=3 mamy: $K = \sum_{j=1}^{3} \omega_j K_j$, *gdzie* K_j *jest wiedzą o lokalizacji celi w* TAI_j *i* ω_j *jest względną ważnością* TAI_j *obu dowódców* $\left(\sum_{j=1}^{3} \omega_j = 1\right)$. *Średnia wiedza wynosi*

$K = \frac{1}{3} \cdot \sum_{j=1}^{3} K_j$. *Metryka wiedzy w tym przypadku jest średnią wiedzą ważoną nad TAIs i przedstawia poziom sytuacyjnej świadomości dowódcy.*

Wiedza jest wartością średnią wiedzy oddziału $K_B = \frac{1}{m} \sum_{i=1}^{m} K_{B,i}$ *i* $K_R = \frac{1}{s} \sum_{i=1}^{s} K_{R,i}$.

Wiedza relatywna lub relatywna świadomość sytuacyjna jest definiowana jako iloraz:

$$\Gamma = \frac{K_B}{K_R}, \; K_R \neq 0.$$

Uwaga.

K_B *jest nieograniczone z góry, zaś* K_R *jest ograniczone z dołu przez 0.*

Przewaga informacyjna

Oczekuje się, że armia wieku informacyjnego będzie w stanie w pełni wykorzystać możliwości jakie drzemią w przewadze informacyjnej. Rodzi się nowe pytanie – jaki wpływ może wywołać przewaga informacyjna na sposób dowodzenia w przyszłości? Amerykańska wizja armii (dokument Army Vision 2010) opiera się na pojęciu przewagi informacyjnej. Przewaga informacyjna definiowana jest jako zdolność do gromadzenia, przetwarzania i rozpowszechniania nieprzerwanego potoku informacji, przy pozbawieniu tych samych możliwości przeciwnika. Zdefiniowanie pojęcia wiedzy relatywnej umożliwia nam dokonanie oceny przewagi informacyjnej pomiędzy Czerwonymi (R) i Niebieskimi (B).

Tablica 8. Możliwe reakcje

Jeśli	***Wtedy***	***I***
$K_B > K_R$	$\Gamma > 1$	Niebiescy posiadają przewagę informacyjną
$K_B < K_R$	$\Gamma < 1$	Czerwoni posiadają przewagę informacyjną
$K_B = K_R$	$\Gamma = 1$	Brak przewagi informacyjnej

Źródło: R. Darilek i inni, Miary efektywności dla armii wieku informacyjnego, Centrum Rand Arroyo, 2002 – tłumaczenie z języka angielskiego, http//www.rand.org/organization/ard/

Dominacja informacyjna

Obok pojęcia przewagi informacyjnej, definiuje się pojęcie dominacji informacyjnej. Pojęcie dominacji informacyjnej nie jest sprecyzowane tak jasno jak pojecie przewagi. „Dominacja informacyjna" jest osiągana wtedy, kiedy „różnica" pomiędzy Niebieskimi i Czerwonymi jest wystarczająco duża. Wynika stąd, że dominacja informacyjna jest osiągana wówczas, gdy przekraczana jest pewna wartość progowa. Metryka wiedzy relatywnej może też być użyta do definiowania dominacji informacyjnej poprzez określenie wartości Γ.

Rozpatrzmy przedział $0 < \beta < 1$ w którym będziemy mieli dominację informacyjną. Niebiescy posiadają przewagę informacyjną, kiedy $K_B > K_R$ (K_R musi być wartością na tyle dużą aby można było uznać, że przeciwnik jest w stanie osiągnąć przewagę informacyjną)[16], wtedy $1 \geq K_B - K_R \geq \beta$. Dzieląc obie strony przez K_R otrzymujemy:

$$\frac{1}{K_R} \geq \Gamma - 1 \geq \frac{\beta}{K_R}.$$

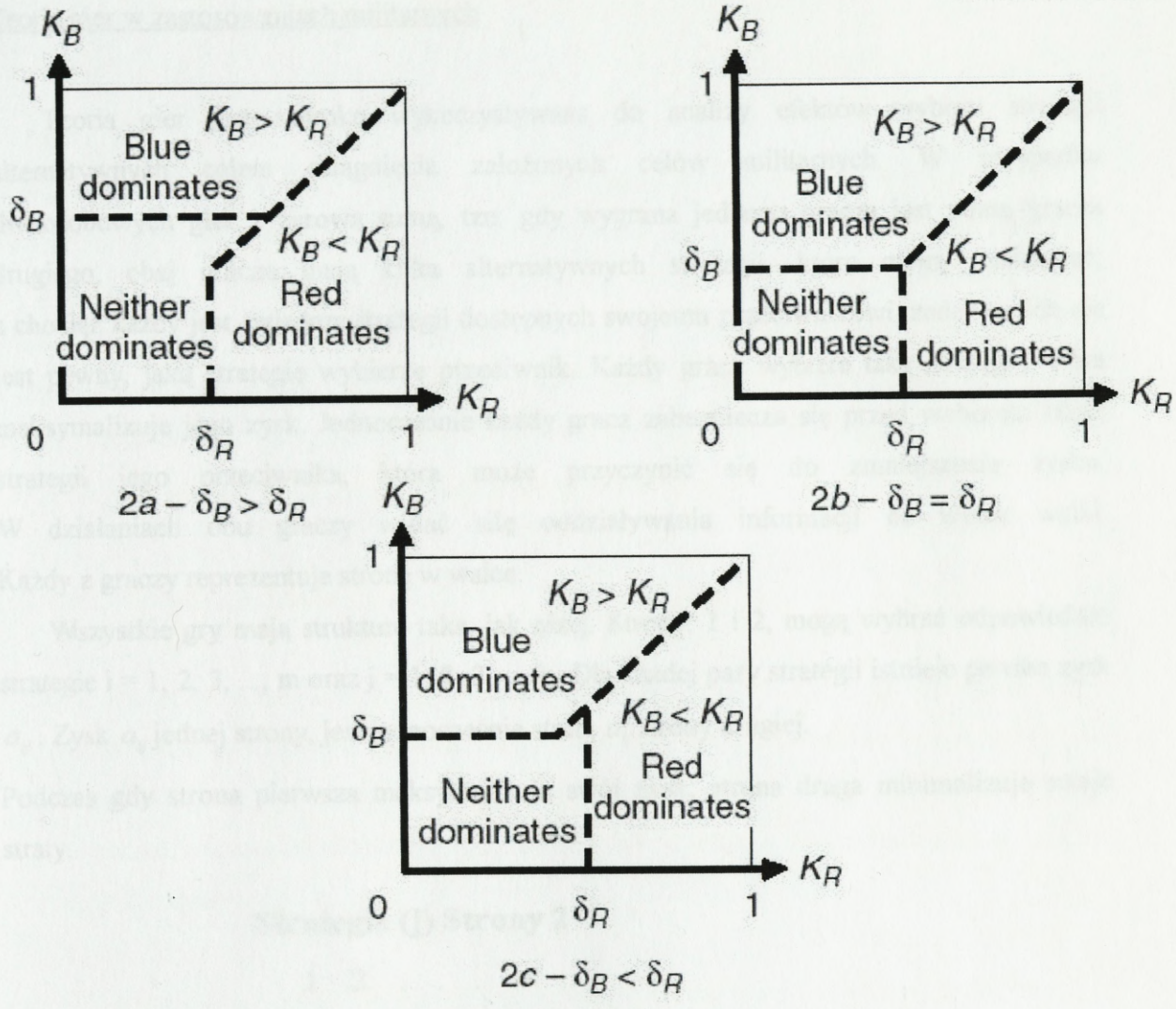
Daje to ograniczenie Γ do przedziału:

$$1 + \frac{\beta}{K_R} \leq \Gamma \leq 1 + \frac{1}{K_R}.$$

Jeśli interesowałyby nas warunki przy których czerwoni uzyskaliby przewagę informacyjną, konieczne byłoby przeprowadzenie podobnego rozumowania.

Graficzne zobrazowanie dominacji informacyjnej przedstawia powyższy rysunek. Zobrazowanie to pozwala zrozumieć różnicę pomiędzy przewagą informacyjną a dominacja informacyjną. Na uwagę zasługuje fakt, że pomimo dominacji jednej ze stron, możliwa jest nadal przewaga informacyjna drugiej strony. Wszystko zależy od okoliczności i charakteru dominacji informacyjnej, bądź przewagi informacyjnej.

[16] Jeśli K_R jest bardzo małe, to nie można mówić o przewadze informacyjnej a w konsekwencji nie można mówić o uzyskaniu dominacji informacyjnej.



Rys.3. Dominacja informacyjna

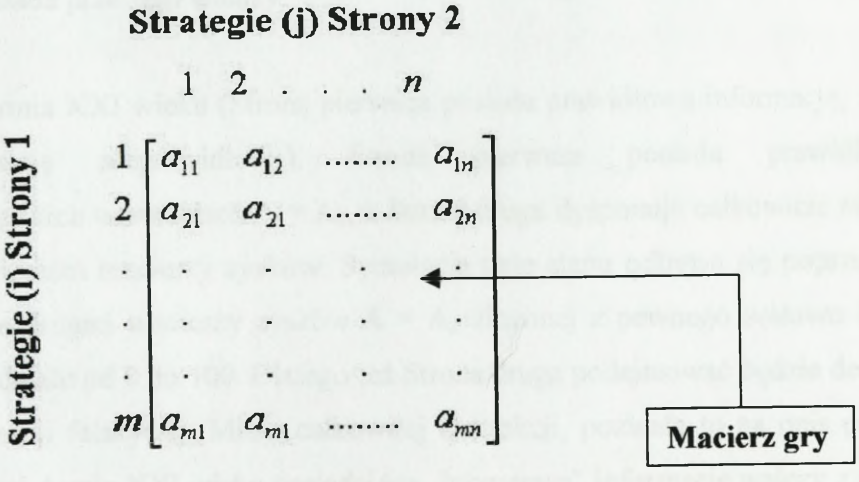
Źródło: R. Darilek i inni, Miary efektywności dla armii wieku informacyjnego, Centrum Rand Arroyo, 2002 – tłumaczenie z języka angielskiego, http//www.rand.org/organization/ard/

Sytuacja taka może powstać w różnych regionach. Oznacza to, że można mówić o jednoczesnej dominacji niebieskich i czerwonych, ale dla dwóch różnych regionów.

Teoria gier w zastosowaniach militarnych

Teoria gier jest szeroko wykorzystywana do analizy efektów wyboru strategii alternatywnych celem osiągnięcia założonych celów militarnych. W przypadku dwuosobowych gier z zerową sumą, tzn. gdy wygrana jednego gracza jest stratą gracza drugiego, obaj gracze mają kilka alternatywnych strategii, które mogą realizować, a chociaż każdy jest świadom strategii dostępnych swojemu przeciwnikowi, żaden z nich nie jest pewny, jaką strategię wybierze przeciwnik. Każdy gracz wybiera taką strategię, która maksymalizuje jego zysk. Jednocześnie każdy gracz zabezpiecza się przed wyborem takiej strategii jego przeciwnika, która może przyczynić się do zmniejszenia zysku. W działaniach obu graczy widać siłę oddziaływania informacji na wynik walki. Każdy z graczy reprezentuje stronę w walce.

Wszystkie gry mają strukturę taką, jak niżej. Strony: 1 i 2, mogą wybrać odpowiednio strategie i = 1, 2, 3, ..., m oraz j = 1, 2, 3, ..., n. Dla każdej pary strategii istnieje pewien zysk a_{ij}. Zysk a_{ij} jednej strony, jest jednocześnie stratą a_{ij} strony drugiej. Podczas gdy strona pierwsza maksymalizuje swój zysk, strona druga minimalizuje swoje straty.



Przypadki zmiennej wiedzy

Przykład

O wojnie moglibyśmy myśleć abstrakcyjnie w następujący sposób. W dowolnej walce wybór strategii Strony 1 będzie miał zasadniczy wpływ na wynik, podobnie jak dla Strony 2. W zależności od warunków walki (stosunki sił, teren, itp.), możliwe strategie mogą powodować większe lub mniejsze różnice. Podobnie do powszechnie uwzględnianych (wyżej wymienionych) warunków walki, należy uwzględnić wartość informacji. Można rozważyć pewne abstrakcyjne przypadki, w których wybór strategii ma bardzo zróżnicowane konsekwencje w odniesieniu do uzyskanych wyników.

Możliwe sa cztery przypadki:

- Gra 1: dla armii współczesnej – przypadek klasyczny (obie strony posiadają prawidłową wiedzę). Strona pierwsza i Strona druga mają wspólną i prawidłową wiedzę w postaci wszystkich wartości z macierzy zysków **A**. Obie strony mają tę samą informację o zyskach, ale nie są świadome, jakich wyborów dokonuje przeciwnik. Żadna ze stron nie posiada przewagi wiedzy.

- Gra 2: Armia XXI wieku (Strona pierwsza posiada prawidłową informację, a Strona druga informację nieprawidłową). Strona pierwsza posiada prawidłową wiedzę o wszystkich wartościach $\mathbf{A} = \mathbf{A}_1$, a Strona druga dysponuje całkowicie *nieprawidłowym* rozumieniem macierzy zysków. Symulacja tego stanu odbywa się poprzez dostarczenie Stronie drugiej macierzy zysków $\mathbf{A} = \mathbf{A}_2$ złożonej z pewnego zestawu liczb losowych z przedziału od 0 do 100. Dlatego też Strona druga podejmować będzie decyzje oparte na informacji fałszywej. Mimo całkowitej abstrakcji, pozwala to na opis pewnej sytuacji, w której Armia XXI wieku posiadająca „luksusową" informację walczy z przeciwnikiem, który nie tylko nie posiada ważnej informacji, ale także jest całkowicie wprowadzony w błąd. Sytuacja ta może być traktowana jako przypadek, w którym Niebiescy (Stroma pierwsza) posiadają przewagę informacyjną.

- Gra 3: AAN (Strona pierwsza posiada prawidłową informację. Strona druga posiada prawidłową informację. Strona pierwsza zna wybór Strony drugiej). Strona pierwsza i Strona druga posiadają prawidłową wiedzę o wartościach **A**, podobnie jak w Grze 1. Strona druga wybiera swoją strategię minimax j^* z prawidłowej macierzy **A**. Jednakże, Strona pierwsza zna wybór Strony drugiej, i zamiast dokonywać wyboru swojej strategii maximin (i), skupia się raczej, jedynie na zyskach odpowiadających wyborowi minimax Strony drugiej i maksymalizuje swój zysk. Symuluje to przypadek, w którym Strona pierwsza posiada perfekcyjny wywiad, a w wyniku, inny rodzaj lub wyższy poziom przewagi informacyjnej. Mimo, że informacja bazowa Strony drugiej w tym przypadku (w przeciwieństwie do Gry 2) nie jest zła, to jest ona wyraźnie gorsza w porównaniu do Strony pierwszej.

- Gra 4: AAN (Strona pierwsza posiada informację prawidłową. Strona druga posiada informację nieprawidłową, Strona pierwsza zna wybór Strony drugiej). W grze czwartej Strona pierwsza posiada prawidłową wiedzę o wszystkich wartościach $A = A_1$, a Strona druga dysponuje całkowicie niepoprawną macierzą zysków $A = A_2$, złożoną z drugiego zestawu liczb losowych z przedziału między 0 i 100, jak w Grze 2. Strona druga wybiera swoją strategię minimax.j* w oparciu o nieprawidłową informację z A_2. Strona pierwsza zna wybór Strony drugiej. Zamiast wykorzystywać swoją strategię maximin, skupia się ona jedynie na zyskach odpowiadających wyborowi minimax Strony drugiej dokonanym w oparciu o nieprawidłową informację i dokonuje swojego wyboru z prawidłowej macierzy A_1. Strona posiada informację perfekcyjną (wiedza maksymalna). Może ona nawet tworzyć swoją pozycję (wykorzystując ofensywne operacje informacyjne) przez aktywną znajomość faktu, że Strona druga posiada złą informację. Zatem Strona pierwsza posiada nie tylko przewagę informacyjną ale także dominację informacyjną.

Z gier tych wynika wyraźny wniosek, że przewaga i dominacja informacyjna pochodzi z dynamicznych interakcji między obu stronami. Mogą się one zmieniać w czasie - np. w trakcie trwania konfliktu. Stąd też powinniśmy myśleć o przewadze i dominacji informacyjnej w kategoriach dynamicznych.

Macierze gier

Porządkując możliwe wybory jednej strony w poziomie a drugiej w pionie, uzyskujemy macierz w układzie Strona 1 – Strona 2. Wpisy w tej macierzy stanowią zyski obu stron (zysk Strony 1 stanowi jednocześnie stratę strony 2). Zawartość macierzy stanowią liczby będące wynikiem symulacji działań obu stron, zależnie od przyjętego wariantu postępowania. W pierwszym uproszczeniu, liczby te mogą być przyjęte jako stałe. Następnie, w oparciu o zastosowany model walki, generowane są kolejne wartości będące rozwiązaniami układu równań różniczkowych. Modelowe równania różniczkowe mogą mieć postać deterministyczną, bądź też mogą być „uzupełnione" o czynniki losowe, stając się równaniami stochastycznymi.

Strona 1 ↓

	Opcja 1	Opcja 2	Opcja 3	Opcja 4
Opcja 1	7	4	7	2
Opcja 2	8	12	16	13
Opcja 3	1	0	4	6
Opcja 4	11	7	4	5

↑ Strona 2

Rys.4. Przykładowa macierz gry.

Źródło: opracowanie własne.

Stochastyczne równania różniczkowe są naturalnym uogólnieniem równań różniczkowych zwyczajnych. Rozwiązując układy stochastycznych równań różniczkowych otrzymujemy tzw. „portrety fazowe". Analiza obserwowanych atraktorów przedstawiających graficzną reprezentację n - rozwiązań danego układu równań, pozwala na wyciągnięcie wniosków co do rozwiązań osobliwych oraz roli atraktorów.

Stochastyczne równanie logistyczne

Funkcja logistyczna jest jedną z podstawowych funkcji trendu i zajmuje wyjątkowo uprzywilejowane miejsce wśród bardzo wielu różnych postaci funkcji używanych do opisu zjawisk ekonomicznych i przyrodniczych. Wynika to stąd, że wzrost organizmów zwierzęcych i roślinnych, liczebność organizmów żywych bytujących w określonych warunkach, jak i produkcja całych gałęzi przemysłu, przebiegają w sposób dający się opisać krzywą logistyczną. Jej równanie można uzyskać na drodze dedukcyjnej wychodząc z rozważań fizykalnych. Równaniem krzywej logistycznej możemy opisać wszystkie zjawiska, których prędkość rozwoju jest wprost proporcjonalna do iloczynu osiągniętego wzrostu nazywanego czynnikiem rozpędu, oraz do odległości tego wzrostu od poziomu nasycenia (a-y), który możemy nazwać czynnikiem hamowania, gdyż z upływem czasu wartość tego czynnika maleje.

Otrzymujemy równanie różniczkowe noszące nazwę prawa wzrostu Robertsona:

$$\frac{dy}{dx} = ky(a-y),$$

gdzie: $k > 0$ jest współczynnikiem proporcjonalności,

a jest poziomem nasycenia (rzędną asymptoty krzywej logistycznej).

Postać funkcji logistycznej (zależna od wartości jej parametrów) wykorzystywana jest między innymi w ekonometrii do wyznaczania trendu logistycznego.
Rozdzielając zmienne w powyższym równaniu otrzymujemy:

$$\left(\frac{1}{y} + \frac{1}{a-y}\right)dy = akdx,$$

co po rozwiązaniu względem y prowadzi do wzoru:

$$y = \frac{a}{1+e^{-akx-C_1}},$$

gdzie C_1 jest dowolną stałą.[17]

Podstawiając $c = ak$ oraz $b = e^{-C_1}$ otrzymujemy równoważną postać krzywej logistycznej:[18]

$$y = \frac{a}{1 + be^{-\alpha x}}.$$

Powyższa funkcja jest nieliniową funkcją trzech parametrów a, b, c.[19] Można wykazać, że funkcja jest stale rosnąca, najpierw w tempie coraz bardziej przyspieszonym $(y' > 0)$, a następnie w tempie malejącym $(y'' < 0)$, aż do niemal całkowitego zahamowania wzrostu, o czym świadczy fakt, że krzywa zbliża się asymptotycznie do prostej $y = a$, co określa się stwierdzeniem, że z biegiem czasu krzywa logistyczna gaśnie. Wynika stąd, że obserwowany gasnący wzrost badanego procesu zmierza do stanu stagnacji.[20]

Ustochastycznienie (randomizacja) modelu może polegać na przyjęciu założenia, że zamiast dodatniego parametru a w równaniu: $\frac{dy}{dx} = ky(a - y)$, uwzględnimy wyrażenie $a + \sigma \cdot \xi_t$, gdzie $\{\xi_t : t \geq 0\}$ jest losowym białym szumem gaussowskim.

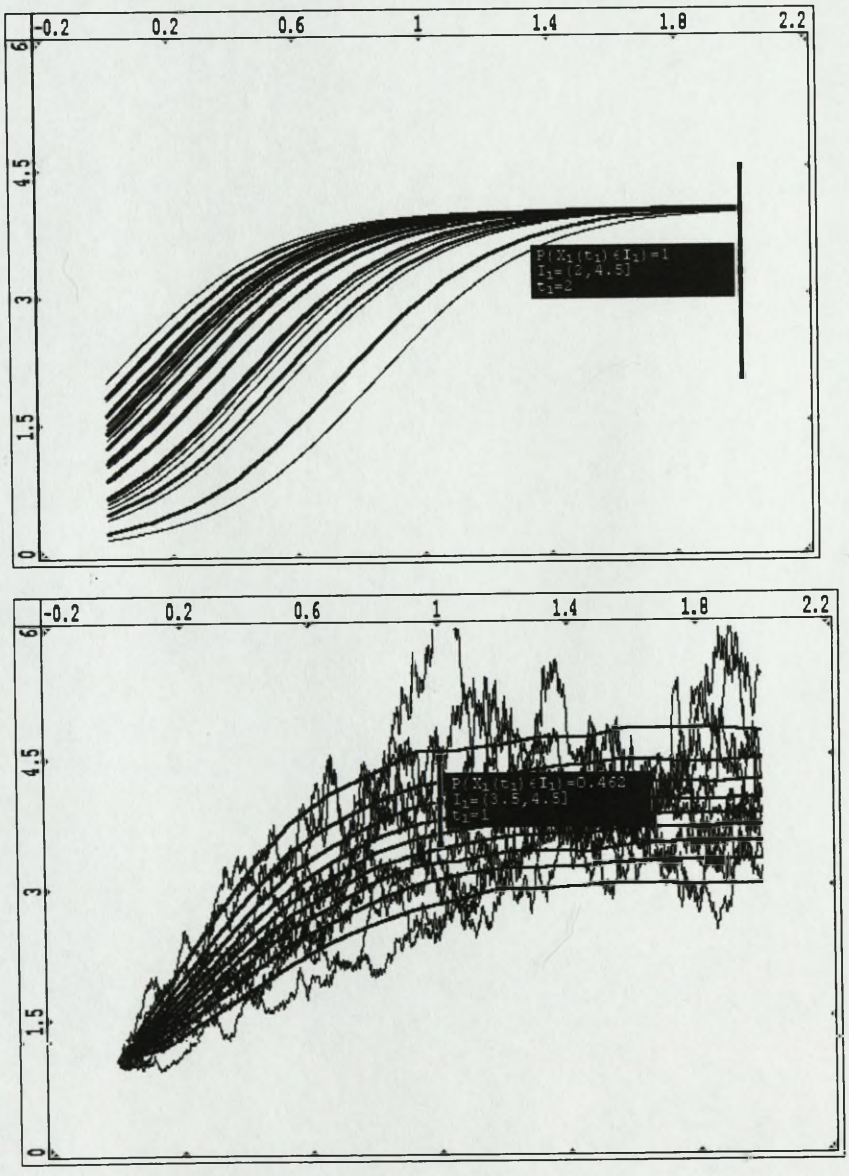
---

[17] S. Smolik, Wyznaczanie parametrów krzywej logistycznej, Przegląd statystyczny R.XXXII- zeszyt 4-1985, s.365 – 373.

[18] H.Hotelling, Differential Equations Subject to Errors and Population Estimates, Journal of the American Statistical Association, 1927 s. 283 –292.

[19] Zainteresowanie krzywą logistyczną sięga końca XIX wieku. Pionierem był P. F. Verhulst, a pierwszym propagatorem jej wykorzystania R. F. Pearl. To on nadał jej ostateczną postać stosowaną z dużym powodzeniem do dzisiaj. Demografowie (np. A.J. Lotka) i biolodzy jako pierwsi próbowali posłużyć się krzywą logistyczną do wyrażenia prawa rozwoju populacji. Krzywą logistyczną można dobrze dopasować do szeregu czasowego obrazującego ilościowy wzrost organizmów żywych w ustalonych warunkach (określony zapas żywności lub określony dopływ żywności). Krzywa ta może również dobrze opisywać wzrost ilościowy organizmów żywych wszędzie tam, gdzie nie nastąpiła jeszcze ingerencja człowieka w ich życie, np. w rezerwatach ścisłych czy niedostępnych obszarach. Wzrost populacji ludzkiej nie rozwija się dokładnie według tej krzywej. Wpływa na to wiele warunków społeczno-ekonomicznych oddziałujących na zasoby żywności. Podważyło to możliwość wykorzystania krzywej logistycznej jako uniwersalnego prawa rozwoju populacji. Wykazano, że rozwój ludności Szwecji przebiegał zgodnie z krzywą logistyczną. Rozwój ten w ciągu ostatnich dwóch stuleci odbywał się prawie w izolacji (nie oddziaływały na niego w znacznym stopniu czynniki zewnętrzne).Funkcja logistyczna dobrze opisuje popyt na dobra trwałego użytku, np. na samochody w krajach wysoko rozwiniętych, na telewizory , na motocykle i motorowery, na motocykle i skutery, na radioodbiorniki; dobrze obrazuje również wzrost liczby abonentów telefonicznych w Polsce. W Polsce zastosowano funkcję logistyczną do badania popytu na dobra trwałego użytku. Na podstawie danych dotyczących stanu posiadania motocykli i rowerów w gospodarstwach wiejskich w Polsce w latach 1951-1962 dokonano ekstrapolacji powstałej funkcji na lata 1963-1966. Podkreślono przy tym, że funkcja logistyczna może dawać dobre prognozy krótkoterminowe i średnioterminowe, natomiast prognozy długoterminowe są ryzykowne. Za pomocą funkcji logistycznej możemy opisać zależność wydajności pracy od stażu pracy na ustalonym stanowisku w przypadku pracowników, którzy nie przekroczyli wieku 50 lat (po przekroczeniu tej granicy wieku wydajność spada). Warto podkreślić, że przy powyższej aproksymacji pomijamy wpływ postępu technologicznego. Krzywa logistyczna dobrze opisuje zmiany szybkości pewnych zjawisk chemicznych, np. szybkości rozpuszczania się soli w wodzie. Podobnie, zależność wielu reakcji od czasu ich trwania, zwłaszcza reakcji syntezy, np. reakcji wody z karbidem, w wyniku której otrzymujemy acetylen. W. Winkler zaproponował użycie krzywej logistycznej do opisu zmian gęstości sieci dróg kolejowych na określonym terenie. Funkcję logistyczną zastosowano w medycynie (np. rozwój pewnych chorób w organizmie) i farmakologii (sposób dawkowania niektórych leków). Podjęto również próby wykorzystania krzywej logistycznej w rehabilitacji i antropologii.

W ten sposób uzyskujemy probabilistyczny model w postaci **SRR** (**S**tochastycznego **R**ównania **R**óżniczkowego). Uzyskane wyżej równanie nazywa się gaussowskim stochastycznym równaniem Verhulsta. Graficzne zobrazowanie problemu, w przypadku deterministycznym i stochastycznym, przedstawia rys.5. Poniższe wyniki uzyskano za pomocą pakietu komputerowego SDE-SOLVER.[21] W przyszłości możliwe będzie wykorzystanie tego pakietu do prac badawczych nad postacią stochastycznego modelu przewagi informacyjnej.



Rys.5. Gaussowskie stochastyczne równanie Verhulsta.
Źródło: opracowano na podstawie: A.Janicki, A.Izydorczyk, Komputerowe metody w modelowaniu stochastycznym, WNT, Warszawa 200 .

---

[20] L.Fahrmeir, G.Tutz, Multivariate Statistical Modelling Based on Generalized Linear Models, Springer, New York 2001, s.403-404.
[21] A.Janicki, A.Izydorczyk, Komputerowe metody w modelowaniu stochastycznym, WNT, Warszawa 2001.

3. ZASTOSOWANE NARZĘDZIA INFORMATYCZNE

Delphi, jako produkt firmy Borland, bazuje na języku programowania Object Pascal. Jest to ewolucja języka programowania stosowanego od wielu lat w tworzeniu aplikacji dla systemu DOS (Disc Operating System). Firma Borland jest głównym producentem i dystrybutorem kompilatorów tego języka na świecie, do których można zaliczyć programy Borland Pascal oraz Turbo Pascal. Wykorzystany w Delphi język programowania Object Pascal jest przykładem doskonałego wykorzystania i adaptacji znanych i uznanych rozwiązań w tworzeniu nowych środowisk, opartych o techniki programowania zorientowanego obiektowo. Oprócz Delphi i Object Pascal'a firma Borland pracuje nad innymi środowiskami programistycznymi. Należą do nich między innymi C++ Builder, JBuilder. Builder C++ jest środowiskiem programistycznym, którego interfejs graficzny oraz idea tworzenia aplikacji jest identyczna z tą którą oferuje Delphi. Builder C++ wykorzystuje również dokładnie tę samą bibliotekę komponentów wizualnych VCL, która to biblioteka jest podstawowym elementem Delphi.

3.1. ZAAWANSOWANE NARZĘDZIA ANALIZY DANYCH DOSTĘPNE W ARKUSZU KALKULACYJNYM EXCEL

DODATEK SOLVER

Solver jest tzw. dodatkiem dostarczanym przez firmę Frontline System (nie jest więc produktem Microsoftu). Narzędzie to świetnie nadaje się do rozwiązywania małych i średnich problemów optymalizacyjnych, przede wszystkim liniowych. Pisząc "małe i średnie" mamy na myśli problemy o niewielkiej liczbie zmiennych decyzyjnych (do 200), bo takie ograniczenie ma standardowy Solver. Firma Frontline System jest producentem różnych komercyjnych wersji Solvera m.in. Premium Solvera, który - dzięki wbudowanemu algorytmowi genetycznemu - radzi sobie z problemami nie do rozwiązania przy użyciu standardowego Solvera. W przypadku gdy polecenie Solver nie jest dostępne w menu Narzędzia, należy zainstalować dodatek Solver.

W polu Komórka celu należy podać adres lub nazwę komórki docelowej. Komórka celu musi zawierać formułę. Aby wartość w komórce była jak największa, należy zaznaczyć opcję **Maks.**, zaś aby wartość w komórce była jak najmniejsza, należy zaznaczyć opcję **Min**. W celu określenia wartość w komórce docelowej, należy zaznaczyć opcję **Wartość** i wpisz wartość w polu obok.

W polu **Komórki zmien**iane podajemy nazwę lub adres każdej komórki zmiennej, oddzielając przecinkami adresy nie przylegających komórek. Komórki zmieniane muszą być bezpośrednio lub pośrednio związane z komórką docelową. Można określić maksymalnie 200 komórek zmienianych. Jeśli chcemy w dodatku Solver automatycznie zaproponować komórki zmieniane dla komórki celu, korzystamy z opcji **Odgadnij**.

W polu **Warunki ograniczające** [na dole strony] podajemy wszystkie ograniczenia, które chcemy zastosować. Wyniki analizy można zapisać w formie raportu.

Cechy i ograniczenia

Możliwe jest podanie maksymalnie 200 zmiennych decyzyjnych

Większość funkcji standardowych (m.in. jeżeli, max, min) nie może być używanych, jeśli zmienne decyzyjne (pośrednio bądź bezpośrednio) są argumentami tych funkcji; czyni to czasem Solver narzędziem dość niewygodnym (trzeba korzystać z różnych trików) lub wręcz niepraktycznym. Wyjaśnienia wymagają zaawansowane opcje Solvera, które dostępne są po naciśnięciu przycisku Opcje.

a) Maksymalny czas

Czas w sekundach, który przydzielamy Solverowi na znalezienie optymalnego rozwiązania.

b) Liczba iteracji

Określa maksymalną liczbę iteracji dla pojedynczego problemu (w każdej iteracji generowane jest nowe rozwiązanie).

c) Dokładność

Określa dokładność spełnienia warunków ograniczających; wartość domyślna 0,000001 oznacza, że np. wartość zmiennej A równa 0,0000001 spełnia warunek ograniczający a>=0.

d) Tolerancja

Wartość standardowa 5% oznacza, że jeśli podczas rozwiązywania problemu Solver znajdzie rozwiązanie o wartości nie odbiegającej od optymalnej bardziej niż o 5%, przyjmie je jako rozwiązanie problemu. Oznacza to, że tak znalezione rozwiązanie nie musi być optymalnym (szczególnie dotyczy to problemów całkowitoliczbowych). Jeśli chcemy mieć pewność, że otrzymane rozwiązanie jest optymalne należy ustawić opcję Tolerancja na 0% (może to znacznie wydłużyć czas obliczeń!).

e) Zbieżność

Parametr określa warunek zatrzymania, jeśli poprawa rozwiązania następuje bardzo powoli. Ściślej biorąc Solver zatrzymuje się, jeśli względna zmiana wartości funkcji celu podczas ostatnich 5 iteracji jest mniejsza niż zadeklarowana wartość parametru.

f) Przyjmij model liniowy

Warto wybrać tę opcję, jeśli funkcja celu i ograniczenia są liniowe, gdyż znacznie przyspiesza to obliczenia (używana jest wtedy metoda Simplex). Należy jednak pamiętać, że Solver testuje, czy założenia liniowości modelu są spełnione i - jeśli tak nie jest - przerywa obliczenia. Co gorsza, nawet jeśli model jest liniowy, ale zmienne bardzo różnią się zakresem wartości, Solver może uznać model za nieliniowy.

g) Przyjmij nieujemne

Jeśli wszystkie zmienne mają podane dolne ograniczenia, opcja ta nie działa. Jednak często zapomina się, że zmienne nie mogą mieć wartości ujemnych (np. wielkość produkcji), bo rozwiązanie traci sens. Wygodnie jest więc zaznaczyć tę opcję, zamiast wpisywać wszystkie dolne ograniczenia typu x>=0.

h) Automatyczne skalowanie

Powinno być wybrane, jeśli ograniczenia lub zmienne w funkcji celu bardzo różnią się zakresem wartości np. jedna podawana w milionach zł, inna - w procentach. Może to spowodować, że Solver nie będzie w stanie znaleźć rozwiązania, szczególnie jeśli wybrano opcję Przyjmij model liniowy.

i) Pokaż wyniki iteracji

Zaznaczamy tę opcję, jeśli po każdej iteracji chcemy widzieć próbne rozwiązanie (pokazuje się ono u dołu arkusza).

j) Estymaty

Określa sposób uzyskania początkowych wartości estymant podstawowych zmiennych w każdym jednowymiarowym procesie poszukiwania. Styczna Wykorzystuje ekstrapolację liniową na podstawie wektora stycznego. Kwadratowa Wykorzystuje ekstrapolację kwadratową. Daje to lepsze wyniki w przypadku zadań wyraźnie nieliniowych.

k) Pochodne

Określa sposób różniczkowania przy wyznaczaniu pochodnej cząstkowej dla funkcji celu i funkcji ograniczeń.

l) Szukanie

Określa metodę używaną w iteracji do wyznaczenia kierunku poszukiwania. Wykorzystuje metodę Newtona, która zwykle wymaga większego obszaru pamięci, ale mniejszej liczby iteracji niż metoda gradientu sprzężonego. Metoda gradientu sprzężonego wymaga mniejszego obszaru pamięci, ale większej liczby iteracji niż metoda Newtona dla zapewnienia tej samej dokładności. Z metody tej należy korzystać w przypadku rozbudowanych zadań i małego obszaru dostępnej pamięci a także, kiedy kolejne iteracje wykazują niewielki postęp.

MOŻLIWOŚCI SOLVERA

Dodatek Solver jest częścią zestawu poleceń czasami zwaną narzędziami analizy typu co-jeśli (analiza typu "co, jeśli?": Proces zmieniania wartości w komórkach, aby zobaczyć, jak te zmiany wpłyną na wyniki formuł w arkuszu).

Na przykład zmienianie stopy procentowej w tabeli amortyzacji, aby określić sumę płatności.). Korzystając z dodatku Solver, można znaleźć optymalną wartość dla formuły (formuła: Sekwencja wartości, odwołań do komórek, nazw, funkcji lub operatorów w komórce, które razem dają nową wartość. Dodatek Solver pracuje z grupą komórek powiązanych, bezpośrednio lub pośrednio, z formułą w komórce docelowej. Dodatek Solver dostosowuje wartości w zmieniających się komórkach określonych przez użytkownika

zwanych komórkami zmienianymi, w celu uzyskania wyniku określonego przez użytkownika na podstawie formuły w komórce docelowej. Użytkownik może zastosować ograniczenia do komórek dostosowanych, komórki docelowe i innych komórek, które są bezpośrednio lub pośrednio związane z komórką docelową. Dodatku Solver można używać do ustalenia maksymalnej lub minimalnej wartości określonej komórki przez zmianę innych komórek, na przykład można zmienić przewidywany budżet reklamowy i zobaczyć wpływ tej zmiany na przewidywany zysk.

Przykładowe arkusze dostępne w dodatku Solver

Do programu Microsoft Excel dołączono przykładowe arkusze znajdujące się w skoroszycie Solvsamp.xls umieszczonym w folderze Office\Samples, demonstrujące różnorodne problemy, które można rozwiązywać, korzystając z dodatku Solver.

Przykładowe arkusze w skoroszycie Solvsamp.xls ułatwiają zdefiniowanie problemów. Aby używać dowolnego spośród 6 arkuszy — Asortyment produktów, Trasy przewozu, Obsada stanowisk, Maksymalizacja wpływów, Portfel papierów wartościowych oraz Projektowanie inżynierskie — należy otworzyć skoroszyt, przełączyć się do żądanego arkusza kalkulacyjnego, a następnie kliknąć polecenie Solver w menu Narzędzia. Komórka docelowa, komórki zmieniane oraz ograniczenia dla arkusza zostały już określone.

Algorytm i metody używane przez dodatek Solver

Dodatek Microsoft Excel Solver używa programu nieliniowej optymalizacji Generalized Reduced Gradient (GRG2), który opracowali Leon Lasdon z University of Texas w Austin oraz Allan Waren z uniwersytetu Cleveland State University.

FUNKCJE KONTROLI SOLVERA [22]

SolverAdd

Równoważny do wybrania Solver z menu narzędzi i wybrania przycisku Add w oknie dialogowym parametru Solvera. Ta funkcja dodaje ograniczenia do bieżącego problemu.

VBA Syntax

SolverAdd(CellRef:=,Relation:=,FormulaText:=)

Macro Language Syntax

=SOLVER.ADD(*cell_ref, relation*, *formula*)

---

[22] *http://www.frontsys.com/mlvbaref.htm*

CellRef jest odniesieniem do komórki lub szeregu komórek na aktywnym arkuszu kalkulacyjnym i tworzy lewą stronę ograniczenia.

Relation określa arytmetyczny związek pomiędzy lewą a prawą stroną oraz czy CellRef musi mieć całkowitą wartość wyniku w rozwiązaniu.

Relacje

1	<=
2	=
3	>=
4	Int (CellRef jest zmienną całkowitą)

FormulaText jest prawą stroną zawężenia i często będzie pojedynczą liczbą ale może być też wzorem lub odniesieniem do szeregu komórek.

Jeżeli **FormulaText** jest odniesieniem do szeregu komórek, liczba komórek w tym szeregu musi odpowiadać liczbie komórek w **CellRef**, mimo, że kształt tych obszarów nie musi być taki sam. Na przykład **CellRef** może być rzędem a **FormulaText** kolumną, tak długo jak liczba komórek jest taka sama.

Uwagi:

- Funkcje **SolverAdd**, **SolverChange** i **SolverDelete** odpowiadają przyciskom **Add**, **Change** i **Delete** w oknie dialogowy o nazwie Parametry Solver.
- Każde ograniczenie jest nie powtarzalnie identyfikowane przez kombinację odniesień komórek po lewej stronie i związku po między lewą i prawą storną.

SolverChange

Równoważnik wybrania **Solver** z menu **Tools** i wybrania przycisku **Change**
w oknie dialogowym Parametry Solvera. Funkcja ta zmienia prawą stronę istniejącego ograniczenia.

VBA Syntax

SolverChange(CellRef:=,Relation:=,FormulaText:=)

Macro Language Syntax

=SOLVER.CHANGE(*cell_ref,relation*,*formula*)

Dla wyjaśnienia argumentów i ograniczeń, zobacz **SolverAdd**

Uwagi:

- Jeśli kombinacja **CellRef** i **Relation** nie pasuje do żadnego istniejącego ograniczenia, funkcja powraca do wartości 4 i nic się nie dzieje.

- Żeby zmienić **CellRef** albo **Relation** istniejącego ograniczenia, użyj **SolverDelete**, żeby usunąć stare zawężenie a potem użyj SolverAdd, żeby dodać ograniczenie w wymaganej formie.

SolverDelete

Równoważnik wybrania **Solver** z menu **Tools** i wybrania przycisku **Delete** w oknie dialogowym Parametry **Solvera**. Funkcja ta usuwa istniejące ograniczenie.

VBA Syntax

SolverDelete(CellRef:=,Relation:=,FormulaText:=)

Macro Language Syntax

=SOLVER.DELETE(*cell_ref,relation,formula*)

Dla wyjaśnienia argumentów i ograniczeń, zobacz **SolverAdd**

Uwagi:

- Jeśli kombinacja **CellRef** i **Relation** nie pasuje do żadnego istniejącego ograniczenia, funkcja powraca do wartości 4 i nic się nie dzieje.

Jeżeli ograniczenie zostanie znalezione, będzie usunięte a funkcja wraca do wartości zero.

SolverFinish

Równoważny z wybraniem opcji i kliknięciem na **OK** w oknie dialogowym Wyniki Solvera, które pojawia się gdy proces rozwiązywania jest zakończony. Okno dialogowe nie będzie wyświetlone.

VBA Syntax

SolverFinish(KeepFinal:=,ReportArray:=)

Macro Language Syntax

=SOLVER.FINISH(*keep_final, report_array*)

KeepFinal jest liczbą 1 albo 2 określającą czy zatrzymać czy wyrzucić ostateczne rozwiązanie. Jeśli **KeepFinal** jest równy **1** lub jest pominięty, to wartości końcowego rozwiązania są przechowywane w zmiennych komórkach. Jeśli **KeepFinal** jest równy **2** to wartości ostatecznego rozwiązania są eliminowane i poprzednie wartości zmiennych komórek są odtworzone.

ReportArray jest tablicą argumentów określającą jakie raporty stworzyć, kiedy Solver jest zakończony.

Jeśli ReportArray jest	Microsoft Excel tworzy
1	raport odpowiedzi
2	raport czułości
3	raport limitu

Kombinacja tych wartości tworzy raporty wielokrotne. Na przykład jeśli **ReportArray** =1,2 to MS EXCEL tworzy raport odpowiedzi i raport czułości. Należy używać funkcji **Array** w VBA do stworzenia matrycy stałych wartości.

SolverFinishDialog

Równoważny z wybraniem opcji w oknie dialogowym Wyniki Solvera, które pojawia się gdy proces rozwiązywania jest zakończony. Okno dialogowe będzie wyświetlone a użytkownik będzie w stanie zmienić opcje, które początkowo określił.

VBA Syntax

SolverFinishDialog(KeepFinal:=,ReportArray:=)

Macro Language Syntax

=SOLVER.FINISH?(*keep_final, report_array***)**

Dla wyjaśnienia argumentów, zobacz SolverFinish.

SolverGet

Zwraca informacje o obecnym problemie Solvera. *VBA Syntax*

SolverGet(TypeNum:=,SheetName:=)

Macro Language Syntax

=SOLVER.GET(*typenum,sheet_name***)**

TypeNum jest liczbą określającą typ informacji. Następujące ustawienia są określone w oknie dialogowym Parametry Solvera.

TypeNum	Zwraca:
1	Odniesienie w oknie Ustawienie Komórki lub wartość błędu # N/A jeżeli Solver nie był używany w aktywnym dokumencie.
2	Liczbę odpowiadającej opcji równa się. 1=Max 2=Min 3=Wartość
3	Wartość w oknie Wartość
4	Odniesienie w oknie Przez Zmianę Komórek
5	Liczbę zawężeń
6	Matrycę lewej strony zawężenia w formie tekstu.
7	Matrycę liczb odpowiadającej związkowi pomiędzy lewą a prawą stroną zawężenia. 1 = <= 2 = =

3 = >=

4 = In

Następujące ustawienia są wyszczególnione w oknie dialogowym Opcje Solvera.

TypeNum	Zwraca:
8	Maxymalny czas kalkulacji
9	Maksymalna liczba iteracji
10	Precyzja (jako liczba dziesiętna)
11	Całkowita wartość marginesu błędu (jako liczba dziesiętna)
12	PRAWDA jeśli pole zaznaczenia założonego Modelu Linowego (Assume Limear Mode) lub założonego modelu LP/QP (Assume LP/QP Model) jest odznaczone. Fałsz w innym wypadku.
13	Prawda jeśli pole zaznaczenia POKAŻ CAŁKOWITY WYNIK (Schow Iteration Result) jest zaznaczone. Fałsz w innym wypadku.
14	Prawda jeśli pole zaznaczenia UŻYJ AUTOMATYCZNEGO SKALOWANIA (Use Automatic Scaling) jest zaznaczone. Fałsz w innym wypadku.
15	Liczba odpowiadająca typowi szacunkowemu 1 = funkcja tangens 2 = funkcja kwadratowa
16	Liczba odpowiadająca typowi pochodnych 1 = przednia 2 = centralna
17	Liczbę odpowiadająca typowi przeszukiwania. 1 = Newton 2 = sprzężone

SheetName jest nazwą arkusz kalkulacyjnego, który zawiera problem Solvera.

SolverLoad

Równoważny z wybraniem **Solver** z menu **Tools**, wybraniem przycisku **Options** z okna dialogowego **Parametry Solvera**, i wybraniem przycisku **Load Model** z okna dialogowego Opcje Solvera. Ładuje specyfikację problemu Solvera, które poprzednio zapisano w arkuszu kalkulacyjnym.

VBA Syntax

SolverLoad(LoadArea:=)

Macro Language Syntax

=SOLVER.LOAD(*load_area***)**

LoadArea jest odniesieniem do aktywnego arkusza kalkulacyjnego, do szeregu komórek z których chcemy załadować kompletną specyfikację problemu.

Pierwsza komórka w **LoadArea** zawiera wzór dla Okna ustawienia komórki (**Seet Cell Box**); druga komórka zawiera wzór dla zmiennych komórek; następne komórki zawierają ograniczenia w formie logicznych wzorów. Ostatnia komórka zawiera matrycę wartości opcji Solvera.

SolverOk

Równoważny z wyborem **Solver** z menu **Tools** i określeniem opcji w oknie dialogowym. Parametry **Solvera** określa podstawowe opcje **Solvera**. *VBA Syntax*

SolverOk(SetCell:=,MaxMinVal:=,Valueof:=,ByChange:=)

Macro Language Syntax

=SOLVER.OK(*set_cell, max_min_val, value_of, by_changing***)**

SetCell odnosi sie do **Set Cell box** w oknie dialogowym **Parametry Solvera**.

SetCell musi być odniesieniem do komórki na aktywnym arkuszu kalkulacyjnym.

Jeśli wejdziemy do komórki musimy wprowadzić wartość dla **MaxMinVal**. W składni makro języka, jeśli nie odwołujemy się do odpowiednich komórek, musimy zawrzeć 3 przecinki przed wartością **ByChanging.**

MaxMinVal odnosi się do opcji **Max**, **Min** i **Value Of** w oknie dialogowym **Parametry Solvera**. Używamy tej opcji tylko wtedy gdy stosujemy odniesienie dla **SetCell.**

MinMaxVal	Określona Opcja
1	Maximize
2	Minimize
3	Match specific value

ValueOf jest liczbą, która staje się celem dla komórki w **Set Cell box** jeśli **MaxMinVal** równa się 3.

ValueOf jest ignorowane jeśli komórka jest maksymalizowana lub minimalizowana.

ByChanging wskazuje zmienne komórki, jeśli wejdziemy w okno **By Changing Cells**. **ByChanging** musi być odniesieniem komórki (zazwyczaj szeregu komórek lub wielokrotnym odniesieniem) na aktywnym arkuszu kalkulacyjnym.

SolverOkDialog

Polecenie to jest równoważne z wybraniem **Solver** z menu **Tools** i określeniem opcji w oknie dialogowym Parametry Solvera. Okno dialogowe Parametry Solvera będzie wyświetlone, a użytkownik będzie w stanie zmienić opcje, które początkowo określono.

VBA Syntax

SolverOkDialog(SetCell:=,MaxMinVal:=, Valueof:=, ByChange:=)

Macro Language Syntax

=SOLVER.OK? (*set_cell, max_min_val, value_of, by_changing***)**

SolverOptions

Równoważnik z wybraniem **Solver** z menu **Tools** i wybraniem przycisku **Options** w oknie dialogowym **Parametry Solvera**. Określa opcje algorytmiczne **Solvera**.

VBA Syntax

SolverOptions(MaxTime:=, Iterations:=, Precision:=,AssumeLinear:= StepThru:=, Estimates:=, Derivatives:=, SearchOption:=, IntTolerance:=, Scaling:=)

Macro Language Syntax

=SOLVER.OPTIONS(*max_time, iterations, precision, assume_linear, step_thru, estimates, derivatives, searchoption, int_tolerance, scaling***)**

Argumenty odnoszą się do opcji w oknie dialogowym Opcje Solvera. Jeśli argument jest pominięty, Solver podtrzymuje bieżące ustawienia tej opcji. Jeśli jakieś argumenty są złego typu, funkcje przywraca wartość błędu #N/A. Jeśli wszystkie argumenty są prawidłowego typu, ale argument ma nieważną wartość, funkcja przywraca dodatnią całkowitą odnoszącą się do jej pozycji. Zero wskazuje, że wszystkie opcje zostały zaakceptowane.

MaxTime musi być całkowitą większą niż 0. Odnosi się do okna MaxTime.

Iterations musi być całkowitą większą niż 0. Odnosi się do okna Iterations

Precision musi być liczbą pomiędzy 0 i 1 ale nie równą 1lub 0. Odnosi się do okna Precision.

AssumeLinear jest wartością logiczną odnoszącą się do okna zaznaczenia Assume Linear Model. Jeśli FAŁSZ, Solver użyje GRG nieliniowego rozwiązania. Jeśli PRAWDA, użyj metody Simplex liniowego rozwiązania.

SolverReset

Równoważny z wyborem **Solver** z menu **Tools**, i wybraniem przycisku **Reset All** w oknie dialogowym **Parametry Solvera**. Usuwa wszystkie wybory komórek i ograniczeń w oknie dialogowym **Parametry Solvera**, i odtwarza wszystkie ustawienia w oknie dialogowym **Opcje Solvera** do ich wartości domyślnej. Funkcja **SolverReset** jest automatycznie wykonywana jeśli wywołana została komórka **SolverLoad**.

VBA Syntax

SolverReset()

Macro Language Syntax

=SOLVER.RESET()

SolverSave

Równoważny z wyborem **Solver** z menu **Tools**, i wybraniem przycisku **Option** z okna dialogowego **Parametry Solvera** i wybrania przycisku **Save Model** w oknie dialogowym **Opcje Solvera**. Zapisuje specyfikację problemu w arkuszu kalkulacyjnym.

VBA Syntax

SolverSave(SaveArea:=)

Macro Language Syntax

=SOLVER.SAVE(*save_area***)**

SaveArea jest odniesieniem na aktywnym arkuszu kalkulacyjnym do szeregu komórek lub do lewego górnego rogu kolumny komórek do których zapiszesz bieżącą specyfikację problemu.

Uwagi:

- Jeśli określimy tylko jedną komórkę dla **SaveArea**, obszar jest rozszerzony w dół na tak wiele komórek jak to jest niezbędne. Żeby utrzymać specyfikację problemu (3 plus liczba zawężeń).
- Jeśli określimy więcej niż 1 komórkę a obszar jest za mały na rozwiązanie danego problemu, specyfikacje problemu nie będą zapisane, a funkcja powróci do wartości 2.
- **SaveArea** musi być na aktywnym arkuszu kalkulacyjnym, ale nie musi być aktywnie zaznaczona.

SolverSolve

Równoważny z wyborem **Solver** z menu **Tools** i wybraniem przycisku **Solver** z okna dialogowego **Parametry Solvera**. Jeśli obliczenia zostaną zakończone sukcesem, funkcja ta przywraca całkowitą wartość wskazującą warunki, które spowodowały, zatrzymanie pracy Solvera.

VBA Syntax

SolverSolve(UserFinish:=, ShowRef:=)

Macro Language Syntax

=SOLVER.SOLVE(*user_finish, show_ref*)

UserFinish jest logiczna wartością określającą czy pokazywać standardowe okno dialogowe **Wyniki Solvera.**

- Jeśli **UserFinish** jest PRAWDĄ, **SolverSolve** przywraca swoją całkowitą wartość bez wyświetlania niczego. Funkcja VBA lub macro powinna zdecydować, którą czynność podjąć (na przykład przez zbadanie wartości powrotu lub pokazanie własnego okna dialogowego. To musi wywołać **SolverFinish** w każdym wypadku, tak aby przywrócić arkusz kalkulacyjny do właściwego stanu.
- Jeśli **UserFinish** jest FAŁSZEM lub został ominięty, Solver wyświetla standardowe okno dialogowe **SolverResults**, pozwalając użytkownikowi zatrzymać lub usunąć wartości końcowego rozwiązania i warunkowo stworzyć raporty.

ShowRef jest funkcją VBA lub macro wywoływaną w miejscu wyświetlania okna dialogowego **Show Trial Solution** (Pokaż Próbne Rozwiązanie).

Przykład definiowania i używania argumentu **ShowRef**:

```
sub test
    answer=SolverSolve(True,"ShowTrial")
end sub
function ShowTrial(Reason as integer)
    msgbox Reason
    ShowTrial=true
end function
```

3.2. ŚRODOWISKO PROGRAMISTYCZNE DELPHI

Delphi jest narzędziem do szybkiego tworzenia aplikacji (ang. RAD - Rapid Application Development) dla systemu operacyjnego Windows. Jako środowisko programistyczne, bazuje na popularnym języku programowania wysokiego poziomu Object Pascal, którego kod jest kompilowany do kodu maszynowego. Środowiska typu RAD, których przedstawicielem jest program Delphi pozwalają na bardzo szybkie i efektywne projektowanie interfejsu użytkownika przy pomocy popularnych w środowisku graficznym Windows technik "drag & drop" ("przeciągnij i upuść").

Istotnym elementem środowiska Delphi, usprawniającym proces tworzenia aplikacji pod Windows jest Biblioteka Komponentów Wizualnych (VCL - z ang. Visual Components Library). Komponenty VCL zawarte są w tzw. klasach Object Pascala i stanowią one szkielet każdego programu. Sprawia to, iż projektowanie aplikacji sprowadza się do ulokowania poszczególnych komponentów w pożądanych miejscach okna formularza, a następnie przypisanie tym komponentom określonych właściwości i określonych akcji (zdarzeń), które powinny one realizować.

W sensie wizualnym głównymi elementami środowiska Delphi są:

- główne okno programu z menu głównym, paskiem narzędzi i paletą komponentów,
- inspektor obiektów,
- przestrzeń robocza.

Główne okno IDE programu (Integrated Development Environment) składa się z menu głównego, umożliwiających dostęp do typowych funkcji Windows (pola File, Edit i Help) oraz do specyficznych funkcji związanych z Delphi i realizacją projektu programistycznego. Należą do nich pola Search, View, Project, Run, Component, Database oraz Tools. Drugim elementem okna głównego, również typowym dla wielu aplikacji systemu Windows, jest pasek narzędzi, umożliwiający szybki dostęp do najczęściej stosowanych opcji znajdujących się w menu głównym takich jak otwieranie, zapisywanie dokumenty czy kompilacja projektu. Zawartość paska narzędzi może być zmieniana komponentów zależności od indywidualnych upodobań użytkownika. Paleta komponentów to specyficzny element interfejsu graficznego Delphi. Znajduje się na niej duży wybór podzielonych tematycznie komponentów (np. etykiet, pól edycyjnych, przycisków itp.) o których będzie mowa w dalszej kolejności. Komponenty te, będące samodzielnymi elementami biblioteki, wykonującymi predefiniowane funkcje, stworzone przez twórców

programu, można wykorzystywać w tworzeniu własnych projektów, przenosząc je na formularz przy pomocy myszy. Paleta komponentów składa się z szeregu zakładek na których znajdują się uporządkowane tematycznie poszczególne komponenty. Najważniejszymi z tych zakładek są:

- Standard (znajdują się na niej przyciski, etykiety, pola wyboru, panele, pola edycji itp.),
- Additional (dodatkowe komponenty, będące uzupełnieniem lub rozwinięciem standardowych komponentów),
- Win32 (inne komponenty typowe dla interfejsu graficznego środowiska okienkowego, oraz związane z poruszaniem się po katalogach katalogach systemie plików),
- System (komponenty obsługujące pewne funkcje systemowe - zegar, technikę OLE - Object Linking and Embedding, Media Player itp.),
- Data Access (komponenty związane z dostępem do baz danych - Data Source, Table, Query - obsługa języka zapytań SQL),
- Data Controls (tzw. komponenty wrażliwe na dane, związane z prezentacją danych i realizacją zapytań),
- QReport (komponenty do sporządzania raportów bazodanowych),
- Dialogs (komponenty związane z obsługą standardowych okien dialogowych systemu Windows),
- Win 3.1 (komponenty symbolizujące kontrolki systemu Windows 3.0, 3.1),
- Samples, ActiveX itp. w zależności od wersji programu.

Znajdujący się w lewej części ekranu Inspektor Obiektów jest narzędziem do bezpośredniej modyfikacji właściwości oraz zdarzeń. Inspektor Obiektów podzielony jest na dwie części: Properties (Właściwości) oraz Events (Zdarzenia).

Właściwości oraz zdarzenia to dwie podstawowe cechy określające wszystkie komponenty - ich wygląd, zachowanie oraz reakcję systemu operacyjnego na ich działanie.

Właściwości (ang. Properties) określają wygląd, położenie, typ oraz sposób działania komponentów. Do najbardziej typowych właściwości komponentów należą takie właściwości jak kolor (Color), szerokość (Width), wysokość (Height). Na uwagę zasługuje fakt, iż istnieje szereg właściwości wspólnych dla wszystkich komponentów, do których należą między innymi właściwości wyżej wymienione. Istnieje jednak olbrzymia grupa właściwości specyficzna dla każdego komponentu z osobna. Właściwości charakteryzujące komponenty

można określać bezpośrednio w Inspektorze Obiektów (Object Inspector) ustawiając konkretne wartości przy zaznaczonej właściwości. Ustawianie w ten sposób właściwości komponentów ma jednak charakter statyczny. Większość kontrolek i innego typu przycisków typowych dla systemu Windows ma charakter dynamiczny tzn. ich zachowanie zmienia się w zależności od wykonywania się programu. Jako przykład mogą posłużyć przyciski lub pola menu, które w pewnych sytuacjach mogą funkcjonować, w innych sytuacjach są "zamrożone" i nie ma możliwości ich obsługi (np. niemożliwe jest użycia przycisku "Wklej" do dopóki w schowku nie znajdą się jakieś dane). Zmiany takich właściwości komponentów dokonuje się w kodzie źródłowym programu w procedurach obsługi zdarzeń, które zostaną przedstawione w dalszej kolejności.

Zdarzenia zachodzą w wyniku interakcji komponentu z użytkownikiem lub systemem, gdy użytkownik lub system wykona jakąś akcję dotyczącą danego komponentu. Do najbardziej typowych zdarzeń w środowisku Windows należy np. kliknięcie myszą (OnClick). Jest to przykład zdarzenia wspólnego dla wszystkich komponentów, poza tym istnieje szereg zdarzeń specyficznych dla poszczególnych komponentów. Obsłużenie zdarzenia polega na stworzeniu procedury obsługi zdarzenia (ang. event handler), która wykonuje określone akcje, jeżeli dane zdarzenie wystąpi. Procedury obsługi zdarzeń są to w zasadzie typowe dla Pascala podprogramy o których mowa będzie w dalszej części wykładu. Są one generowane przez bibliotekę VCL automatycznie i programista nie musi martwić się o ich przygotowanie, jedynym zadaniem, które ma do zrobienia jest właściwe stworzenie kodu wewnętrznego procedury, który w prawidłowy sposób zrealizuje postawione przezeń zadanie.

Główną część Delphi IDE stanowi przestrzeń robocza. Stanowi ją dwa podstawowe elementy. Jednym z nich jest projektant formularzy, służący, jak to sama nazwa wskazuje do projektowania formularzy. Formularz w Delphi jest po prostu reprezentacją okna wyświetlanego na ekranie po uruchomieniu skompilowanego programu. Projektant Formularzy umożliwia umieszczanie, przesuwanie i skalowanie komponentów umieszczanych na formularzach. Utworzenie nowego projekt w Delphi wiąże się zwykle z wygenerowaniem przez nie nowego formularza, który od tej pory staje się głównym oknem programu. Nie ma oczywiście problemów z dodawaniem do danego projektu większej liczby formularzy - stają się one zwykle oknami dialogowymi, podrzędnymi w stosunku do formularza głównego.

Drugim elementem przestrzeni roboczej Delphi IDE jest ukryty pod formularzem Edytor Kodu, służący do wpisywania i edycji kodu tworzonego programu, procedur i funkcji wykonujących określone zadania oraz obsługujących określone zdarzenia. Jak to zaznaczono na wstępnie kod źródłowy programu tworzonego w środowisku Delphi jest pisany w języku programowania Object Pascal. Każdemu formularzowi w projekcie przypisane jest okno Edytora Kodu i podobnie jak w przypadku formularzy, jedno z okien Edytora Kodu ma charakter nadrzędny w stosunku do innych.

Wszystkie elementy IDE opisane powyżej, ściśle współpracują ze sobą w procesie tworzenia aplikacji, pomimo faktu, że każde z nich jest zupełnie odrębnym, autonomicznym tworem. Stąd też doniosłe znaczenie podczas pracy nad tworzeniem aplikacji ma umiejętność zarządzania tymi elementami. Umożliwia to bardzo rozbudowany system zarządzania plikami przez środowisko Delphi.

PLIKI UŻYWANE W PROJEKTACH DELPHI

Tworzenie skomplikowanych programów w środowisku Delphi prowadzi często do powstania znacznej liczby elementów różnego typu w obszarze przestrzeni roboczej, co w pewnych sytuacjach może być uciążliwe i w znacznym stopniu utrudniać pracę programiście. Bardzo istotna zatem, w tego typu sytuacjach staje się możliwość zarządzania poszczególnymi elementami niezbędnymi w procesie tworzenia programu. W Delphi sytuację tą rozwiązano w ten sposób, że poszczególne elementy różnego typu mają określoną reprezentację w postaci pliku z rozszerzeniem typowym dla danego elementu. Są to pliki, które można nazwać plikami pomocniczymi, które są nieodłącznymi częściami nieskompilowanych projektów. Do najważniejszych plików o których jest mowa powyżej należą:

- plik źródłowy projektu, zawierający kod wykonywany przy uruchamianiu aplikacji. Plik źródłowy projektu posiada rozszerzenie .dpr,
- plik formularza, zawierający opis danego formularza oraz umieszczonych na nim komponentów. Jest to fizyczna reprezentacja nieskompilowanego formularza, do której odwołuje się Delphi w celu załadowania uprzednio zapisanego formularza. Rozszerzenie pliku formularza jest następujące .dfm,

- plik źródłowy modułu, zwierający kod źródłowy programu w języku Object Pascal (każdy moduł posiada swój plik źródłowy). Rozszerzenie pliku źródłowego modułu brzmi .pas,
- plik opcji projektu o rozszerzeniu .dof,
- binarny plik wynikowy kompilatora (skompilowany moduł) o rozszerzeniu .dcu,
- skompilowany binarny plik zasobów o rozszerzeniu .res,
- wynikowy program wykonywalny o rozszerzeniu .exe.

Wymienione powyżej pliki są najważniejszymi plikami, z którymi programista ma do czynienia w procesie tworzenia każdej aplikacji. Oprócz powyższych plików istnieje w Delphi szereg innych typów plików w zależności od sposobu tworzenia aplikacji. Należą do nich między innymi pliki konfiguracyjne projektu (rozszerzenie .cfg), pliki obszaru roboczego projektu, zawierające informacje na temat wyglądu obszaru roboczego podczas ostatniego zachowywania lub zamykania projektu (rozszerzenie .dsk), pliki grup projektów (.bpg), pliki źródłowe pakietu (.dpk), skompilowane pakiety (.bpl).

Wszystkie typy plików typowych dla środowiska programistycznego Delphi można najogólniej podzielić na dwie kategorie:

- pliki, zawierające dane wejściowe dla aplikacji,
- pliki wynikowe, tworzone przez Delphi podczas różnych etapów kompilacji.

Struktura i zawartość poszczególnych plików (chodzi tutaj głównie o pliki modułowe, nieskompilowane, z kodem źródłowym) przedstawione są w dalszej kolejności.

RODZAJE MODUŁÓW

Każda aplikacja napisana w Delphi (z interfejsem graficznym) posiada kilka plików źródłowych, zwanych modułami. Moduły są to pliki tekstowe, które są w procesie kompilacji tłumaczone na kod maszynowy. Jak to już wyżej wspomniano dla środowiska Delphi podstawowym modułem jest moduł główny projektu (ang. project source unit) o rozszerzeniu .dpr, zawierający tekst źródłowy projektu. Moduł ten jest automatycznie generowany przez Delphi w momencie tworzenia nowej aplikacji. Zwykle nie modyfikuje się ręcznie tego pliku, z wyłączeniem niektórych zaawansowanych technik programistycznych. Ewentualne błędy

powstałe podczas modyfikacji modułu głównego mogą uniemożliwić w ogóle kompilację całego projektu.

Drugim typem modułu, występującym zawsze w aplikacjach Delphi, jest moduł formularza. Moduł ten, jak sama nazwa wskazuje, zawiera kod źródłowy związany z formularzem. Moduły formularzy posiadają rozszerzenie .pas, i są to najczęściej wykorzystywane typy modułów w programach Delphi. W zasadzie z każdym modułem (.pas) nierozerwalnie i bezpośrednio związany jest plik formularza (.dfm). Nie jest to jednak konieczność i zdarza się, że w pewnych sytuacjach możemy spotkać się w Delphi z modułami zawierającymi jedynie kod źródłowy bez odpowiednika graficznego w postaci formularza. Nie możliwe jest jednak istnienie formularza bez modułu tekstowego.

Moduły w Delphi mają pewien określony format (podobnie jak wszystkie inne pliki źródłowe w różnych językach programowania). Jest to niezbędne aby kompilator mógł skompilować kod do postaci wykonywalnej.

Moduł główny

Każdy moduł projektu głównego zawiera na początku typowe dla języka Pascal słowo kluczowe "program" po którym następuje nazwa modułu. Delphi nadaje każdemu nowemu projektowi domyślną nazwę "Project1", jeśli nie zapisze się modułu pod bardziej znaczącą nazwą. W dalszej kolejności elementem każdego modułu jest znana z typowego Pascala sekcja "Uses", w której wymienione są nazwy wszystkich innych modułów, do których są odwołania w tym module. Kolejnym elementem każdego modułu głównego jest dyrektywa kompilatora, nakazująca mu dołączenie pliku zasobów projektu (ang. resources) - {$R *.RES}.

Zasadniczą częścią modułu głównego jest blok zawierający się między słowami kluczowymi "begin" i "end". Znajdują się tam instrukcje odpowiedzialne za inicjalizację aplikacji, utworzenie jej głównego formularza i uruchomienie aplikacji. Jak to już wcześniej zaznaczono nie należy modyfikować pliku modułu głównego ręcznie, gdyż może to prowadzić to powstania błędów, które w rezultacie mogą uniemożliwić kompilację całego projektu.

Moduł formularza

Innym rodzajem modułu typowego dla środowiska programistycznego Delphi jest moduł formularza. Najogólniej można stwierdzić, iż struktura modułu głównego i modułu formularza są do siebie bardzo podobne. Główną różnicą w obu przypadkach jest fakt, iż moduł formularza posiada dwa dodatkowe słowa kluczowe: "interface" i "implementation".

Każdy moduł formularza rozpoczyna się od słowa "unit" (równoważnik słowa kluczowego "program" w module głównym), po którym następuje nazwa modułu. Podobnie jak w przypadku modułu głównego Delphi również nadaje temu modułowi domyślną nazwę do chwili zachowania go pod inną, własną nazwą. Domyślna nazwa modułu formularza brzmi "Unit1".

W sekcji "interface" deklarowane są identyfikatory, które mają być widoczne na zewnątrz modułu (w innych modułach, odwołujących się do tego modułu, czyli mających jego nazwę na swojej liście modułów "uses"). Mogą tutaj znaleźć się deklaracje typów, stałych, zmiennych oraz deklaracje funkcji oraz procedur, które powinny być widoczne również na zewnątrz niniejszego modułu (w innych modułach, jeśli projektowana aplikacja ma interfejs wielookienkowy). Sekcja interface rozpoczyna się od wspomnianej listy "uses", w której wyeksponowane są informacje niezbędne dla Delphi do kompilacji modułu.

W sekcji implementation znajduje się właściwy kod modułu. Sekcja ta zaczyna się słowem kluczowym "implementation", a kończy zwykle słowem kluczowym "end", w wyjątkowych sytuacjach (w modułach posiadających specjalną sekcję inicjalizacyjną) słowem kluczowym "initialization". W sekcji tej (implementation) może się znaleźć, w projektach składających się z więcej niż jednego formularza sekcja uses, w której znajdują się odwołania do wszystkich innych modułów powiązanych z tym modułem. W dalszej kolejności jest miejsce na deklarację typów, stałych oraz zmiennych, lokalnych dla niniejszego modułu, ale widocznych we wszystkich podprogramach będących częściami składowymi niniejszego modułu. Ważnym elementem każdego modułu formularza jest dyrektywa kompilatora {$R *.DFM}, wiążącą plik modułu (*.pas) z plikiem formularza (*.dfm). Moduł formularza nie zaczyna się od słowa kluczowego "begin", w przeciwieństwie do modułu głównego projektu. Kończy się jednak podobnie jak moduł główny projektu słowem kluczowym "end". Bezpośrednio za dyrektywą kompilatora {$R *.DFM} jest miejsce na tworzenie właściwego kodu modułu, czyli krótko mówiąc na tworzenie podprogramów

(funkcji i procedur) będących częściami składowymi niniejszego modułu. Do innych słów kluczowych typowych dla Delphi i jego modułów zaliczamy słowa "const", "var", "type", oznaczające odpowiednio deklarację stałych, zmiennych oraz typów danych, których znaczenie w programowaniu w Delphi zostanie omówione w rozdziale poświęconym językowi programowania Object Pascal.

PROCES KOMPILACJI

Przedstawione w poprzednich rozdziałach typy plików oraz modułów, niezbędnych przy pracy ze środowiskiem programistycznym Delphi związane są bezpośrednio z procesem kompilacji projektów do programów wykonywalnych.

W procesie kompilacji projektu można wyodrębnić kilka etapów. W pierwszej kolejności kompilowane są plik źródłowy projektu (.dpr) oraz wszystkie pozostałe moduły będące częścią projektu (.pas). Efektem tej czynności kompilatora jest utworzenie binarnych plików zasobów (.dcu). W następnym etapie do akcji wkracza konsolidator zwany "linkerem", którego zadaniem jest połączenie owych plików binarnych z potrzebnymi bibliotekami, a następnie generowany jest ostateczny plik wykonywalny (.exe), który może być bezpośrednio wykonywany przez użytkownika z poziomu środowiska Windows.

PROGRAMOWANIE OBIEKTOWE W DELPHI

Programowanie obiektowe jest nowoczesnym stylem programowania, który to styl w znaczącym stopniu przyczynił się do ułatwienia pisania aplikacji dla systemu operacyjnego Windows. Programowanie obiektowe to próba "przeniesienia" rzeczywistości na platformę maszyn liczących - komputery. W przeciwieństwie do tradycyjnego programowania strukturalnego, gdzie programista zmuszony jest do znajomości wielu, a zwykle wszystkich parametrów symulowanego obiektu, programowanie obiektowe daje możliwość wydawania poleceń dla symulowanego obiektu rzeczywistego bez konieczności zagłębiania się w szczegóły jego konstrukcji czy stanu, podobnie jak ma to miejsce w życiu codziennym człowieka, który postrzegając przedmioty (obiekty) w sposób intuicyjny nie zwraca uwagi na ich zachowanie i szczegóły ich konstrukcji. Podstawowym pojęciem związanym z programowaniem obiektowym jest pojęcie **klasy**.

Klasa jest obiektem składającym się z pól i metod (funkcji i procedur Object Pascala) współdziałających w celu wykonania określonego zadania lub funkcji. Za przykład klasy

może posłużyć typowa kontrolka Windows - CheckBox - pole wyboru. Każdy obiekt rzeczywisty posiada trzy elementy - tożsamość (nazwę), stan lub cechy oraz zachowanie się obiektu (odpowiednikiem czego są metody w Object Pascalu). Tożsamość dowolnego obiektu CheckBox określona jest przez jego nazwę, którą może określić sam programista. Stan tego obiektu reprezentują wspominane już *Właściwości* - np. czy obiekt jest zaznaczony, czy też nie. Metody natomiast mogą w przypadku tego obiektu posłużyć do przypisywania konkretnych jego właściwości - ustawianie stanu (zaznaczanie i odznaczanie). Klasy, na których oparty jest Object Pascal, a w szczególności dotyczy to biblioteki komponentów wizualnych VCL posiadają 4 poziomu dostępu do swoich składników:

- **Private** (zawiera wszystkie szczegóły implementacyjne znane tylko twórcy klasy - programiście, który ją stworzył. Pola i metody "prywatne" nie są widziane na zewnątrz klasy),
- **Public** (poziom publiczny zawiera cały interfejs, za pomocą którego świat zewnętrzny komunikuje się z tą klasą),
- **Protected** (poziom elementów chronionych - poziom dostępu rzadko stosowany),
- **Published** (poziom dostępu używany przy pisaniu komponentów - klas będących elementami wielokrotnie wspominanej biblioteki VCL).

Związane jest to z tzw. **hermetyzacją danych**, co oznacza ukrywanie pewnych szczegółów implementacyjnych obiektów przed światem zewnętrznym - jak to wspomniano powyżej - użytkownik obiektu rzeczywistego nie koniecznie musi wiedzieć dokładnie jak dany obiekt funkcjonuje np. do prowadzenia auta w zasadzie nie jest konieczna znajomość wszystkich faz pracy silnika czterosuwego itp. Podobnie jest z programistą - korzystając z obiektów Object Pascala nie koniecznie musi dokładnie wiedzieć w jaki sposób realizują one konkretne zagadnienia programistyczne - a jednak może być w stanie wykorzystać je w swoich programach. Na tej idei właśnie powstała biblioteka VCL - oferuje ona twórcom aplikacji gotowe obiekty - komponenty o cechach wskazanych powyżej (nazwa, stan, działanie). Obiekty te to twory skomplikowane, umożliwiające jednak dość proste ich stosowanie dzięki wspomnianej powyżej hermetyzacji. Nauka programowania obiektowego w języku Object Pascal nie jest celem tej pracy, jednak trudno jest omówić funkcjonowanie i ideę projektowania aplikacji bez pobieżnego przynajmniej omówienia języka programowania, dlatego zasygnalizowane zostaną w tym miejscu najistotniejsze elementy języka programowania wysokiego poziomu jakim jest Object Pascal.

Jedną z metod umożliwiających integrację danych i dokumentów pochodzących z różnych źródeł w jednym miejscu jest zastosowanie technologii **OLE**. Przedstawiamy jej implementacje w różnych narzędziach programistycznych. Technologia OLE (Object Linking and Embeding) jest mechanizmem pozwalającym na osadzanie (składowanie wewnątrz dokumentu) bądź dołączanie (składowanie w osobnym pliku) dowolnych obiektów pochodzących z różnych źródeł do jednego dokumentu. Umożliwia to wykorzystanie możliwości oferowanych przez różne programy bez ich uruchamiania. Przykładem mogą być obiekty pochodzące z Excela osadzone w dokumencie Worda. Nie musimy wywoływać oddzielnej kopii Excela, aby wprowadzić zmiany w arkuszach. Technika łączenia i osadzania obiektów opiera się na dwóch podstawowych elementach: **kontenerach i serwerach OLE.**

Kontener jest szczególnym typem aplikacji, pozwala bowiem w swoich dokumentach osadzać obiekty pochodzące z innych programów. Dokument znajdujący się w aplikacji kontenera musi mieć możliwość wyświetlenia oraz zapamiętania obiektu osadzonego. Serwer OLE zapewnia użytkownikowi możliwość tworzenia dokumentów oraz danych przeznaczonych do osadzenia w kontenerze OLE. Aplikacja OLE może być równocześnie kontenerem oraz serwerem. Za funkcje związane z techniką OLE odpowiedzialne są klasy OleLinkingDoc, OleServerDoc oraz OleDocument. Pierwsza z nich służy do osadzania innych obiektów w dokumencie, druga pozwala na osadzanie edytora w innych programach zaś ostatnia pozwala na składowanie obiektów OLE w jednym dokumencie.

Wśród **Serwerów OLE** wyróżnia się dwa typy: serwer wewnątrzprocesowy (in-process) oraz serwer lokalny (out-of-process). Pierwszy jest specjalną biblioteką DLL. Może on być umieszczony oraz uruchomiony tylko i wyłącznie w kontenerze aplikacji OLE i działa w przestrzeni adresowej programu wywołującego. Natomiast serwer lokalny jest oddzielną aplikacją typu EXE i działa we własnej przestrzeni adresowej, innej niż przestrzeń adresowa aplikacji wywołującej.

W chwili, gdy użytkownik chce umieścić obiekt OLE w aplikacji, mechanizm OLE przegląda listę typów obiektów do osadzania, a następnie wywołuje odpowiednią aplikację serwera. Lista obiektów umieszczona jest w systemowym rejestrze i uzupełniana przy każdym rejestrowaniu aplikacji serwera OLE w systemie. Wpis ten oprócz typu obiektu zawiera nazwę i rozszerzenie pliku, ścieżkę dostępu oraz inne informacje konieczne do lokalizacji i uruchomienia serwera OLE.

Technologia OLE pozwala również na osadzanie obiektów pochodzących z innej maszyny niż lokalna. Oznacza to, że serwer aplikacji znajduje się na maszynie **A**, natomiast kontener, w którym zostaje osadzony na maszynie **B**. Z punktu widzenia aplikacji

kontenera aplikacja serwera zostaje uruchomiona w przestrzeni bieżącego procesu niezależnie od tego, czy jest to ten sam komputer, czy komputer zdalny.

Automatyzacja OLE

Obiekty OLE można przetwarzać na dwa sposoby: przez uaktywnienie obiektu (activation) lub poprzez tzw. OLE Automation. W pierwszym przypadku użytkownik może edytować dokument bezpośrednio w jego własnym środowisku w aplikacji macierzystej zostaje otwarta osadzona część dokumentu i użytkownik może ją zmienić, korzystając ze standardowego interfejsu aplikacji serwera OLE. Drugi przypadek jest nieco bardziej skomplikowany i w ogólności polega na edytowaniu dokumentu osadzonego przy wykorzystaniu interfejsu aplikacji kontenera. Aby mogła zaistnieć taka sytuacja, należy odpowiednio oprogramować serwer automatyzacji lub sterować aplikacją serwera z poziomu aplikacji klienta. Obecnie większość aplikacji dla Windows jest tworzona w postaci serwerów lokalnych OLE Automation.

W programach Word czy Excel jest bardzo mało funkcji, których nie można wywołać metodami automatyzacji, dzięki czemu programiści mogą bez żadnych ograniczeń używać kontrolek dostarczanych przez te aplikacje. Oprócz przetwarzania tekstów i funkcji arkusza kalkulacyjnego aplikacje Microsoftu pozwalają na wykorzystanie wbudowanego wspomagania drukowania oraz systemu wysyłania dokumentów pocztą elektroniczną.

Biblioteka komponentów VCL w Delphi Builder zawiera całą niezbędną funkcjonalność potrzebną do tworzenia aplikacji bazodanowych. Zdarzają się jednak sytuacje, w których trzeba sięgnąć po funkcje niższego poziomu. Mamy wtedy dostępne całe API **B**orland **D**atabase **E**ngine.

BDE oferuje ponad setkę funkcji do zarządzania bazami, tabelami, sesjami, nawigacją i modyfikowaniem danych..

Przykłady składni API BDE, oraz rozwiązania kilku często pojawiających się problemów.

Wymuszenie zapisu

BDE ma własny bufor zapisu rekordów. Może się zdarzyć sytuacja, że dane, które pieczołowicie wpisywaliśmy są trzymane w pamięci i nastąpi zanik napięcia.. Aby wymusić zapis bufora konkretnej tabeli należy użyć funkcji dbiSaveChanges.

Check(dbiSaveChanges(Table.Handle))

Uwaga: Wywołania funkcji API BDE należy otaczać funkcją Check(). Tłumaczy ona błędy zwracane przez BDE na wyjątki Delphi Buildera.

Kasowanie rekordów w tabelach dBase i FoxPro

W momencie usuwania wierszy w tabelach dBase i FoxPro nie SA one fizycznie usuwane z pliku, a tylko zaznaczone jako nieistniejące. Jeżeli w tabeli wykonujemy wiele operacji usuwania i dodawania rekordów to po jakimś czasie plik zaczyna rosnąć. Możemy wtedy wykorzystać pakowanie tabeli:

Check(DbiPackTable(Table.DBHandle, Table.Handle, nil, szDBASE, TRUE))

Efektem ubocznym tej własności (zwanej "soft deletes") jest możliwość przywrócenia usuniętych rekordów:

Check(dbiUndeleteRecord(Table.Handle))

Przyspieszenie indeksów

Przy intensywnym używaniu tabeli indeksy przestają być tak efektywne jak mogłyby być. Istnieje wtedy możliwość ich zregenerowania - dzięki temu zostanie uporządkowany plik z indeksem. Regenerację wszystkich indeksów tabeli wykonujemy operacją:

Check(DbiRegenIndexes(Table.Handle));

Odświeżanie zawartości tabel

W środowisku, w którym wielu użytkowników modyfikuje te same dane wskazane jest sprawdzenie, czy rekordy, które użytkownikowi pokazujemy na ekranie są na pewno aktualne. W komponentach klasy TDataSet jest metoda Refresh, która pozwala na odświeżenie tabel, ale jest sposób na odświeżenie na raz wszystkich oglądanych przez nas danych. Służy do tego bezparametrowa funkcja dbiCheckRefresh. Sprawdza ona, czy są jakieś zmienione rekordy we wszystkich tabelach związanych z aktualną sesją BDE.

EXCEL a DELPHI [23]

Automatyzacja pozwala jednej aplikacji na kontrolę nad inną aplikacją. Aplikacja kontrolowana jest nazywana automatyzowanym serwerem (w naszym przypadku jest to Excel). Aplikacja kontrolująca serwer, jest nazywana kontrolerem automatyzacji. Są dwa sposoby dostępu do automatyzowanych serwerów (Excel – Solver).

---

[23] *http://freespace.virgin.net/graham.marshall/excel.htm*

Sposób pierwszy – późne wiązanie.

Używając tej metody nazwy funkcji i parametry danych są rozwiązywane w czasie wykonania, wszystkie parametry są przepuszczane jako warianty (opcje). Jako, że żadne błędy w nazwach funkcji i w parametrach danych nie są (rejestrowane, ujawniane, zapisywane) w czasie kompilacji – (czyli w czasie przekształcania programu ujętego w język programowy na język maszynowy) ta metoda jest podatna na błędy. Nazwy funkcji i parametry muszą być (sprawdzane), w czasie wykonania. Stąd ta metoda jest wolna. Jedyną zaletą tej metody dla programowania w Delphi jest to że, jest to jedyny sposób gdzie dodatkowe nadobowiązkowe parametry mogą być (ominięte, wyeliminowane) z nazw funkcji.

Sposób drugi – wczesne wiązanie (używanie bibliotek danych)

Używając tej metody nazwy funkcji i parametry danych są rozwiązywane w czasie kompilacji to biblioteka danych musi być (zaimportowana, przeniesiona) do Delphi. Biblioteka danych jest językowym opisem wszystkich przedmiotów i funkcji przedstawianych przez serwer. Wszystkie parametry muszą być dostarczone nawet przy wywoływaniu funkcji gdzie w dokumentacji mówi się, że są nie obowiązkowe.

W związku z przewagą tego sposobu (wczesnego wiązania), reszta tego dokumentu przedstawia podstawy tworzenia aplikacji za pomocą wczesnego wiązania. Wszystkie aplikacje, które używają automatyzacji Excela powinny korzystać z tej techniki chyba, że jest uzasadniony powód rezygnacji z tego sposobu.

4. SYSTEM WSPOMAGAJĄCY PROCES OCENY PRZEWAGI W WALCE I OPERACJI

Współczesny świat charakteryzuje się dużym przepływem informacji, zarówno otrzymywanej w sposób jawny, jak i zdobywanej bez zgody jej właściciela. Umiejętność wykorzystania tych informacji przy podejmowaniu decyzji stanowi bardzo często o sukcesie lub porażce przedsięwzięcia. Ten sam problem stoi przed dowódcami sił zbrojnych w obliczu konfliktu zbrojnego. Tym bardziej umiejętność ta nabiera znaczenia, że porażka w tym przypadku oznacza nie tylko straty materialne ale przede wszystkim ludzkie. Dlatego coraz częściej, wykorzystanie narzędzi informatycznych do przyspieszenia i usprawnienia pracy dowódców staje się obiektem prac w większości państw.

Poniżej zaprezentowane rozważania oparte zostały na matematycznym modelu wyznaczania przewagi jednej ze stron konfliktu zbrojnego. Posłużono się metodą taksonomii numerycznej opartą na algorytmie opracowanym przez Hugo Steinhausa.

Do wykonania modelu systemu wykorzystano metodę TAX[24] służącą obliczaniu stosunku sił stron konfliktu. Głównym zadaniem zaprojektowanego systemu jest dostarczenie użytkownikowi gotowych propozycji rozwiązań dotyczących takiego wykorzystania posiadanych środków walki, aby uzyskać przewagę w starciu zbrojnym z potencjalnym przeciwnikiem. System ten ma generować następujące charakterystyki:

- stosunek sił na początku starcia,
- wykresy zmian stosunku sił w czasie trwania starcia (przy stałych w czasie czynnikach stymulujących działania strony własnej),
- wykresy zmian stosunku sił w czasie trwania starcia (przy zmiennych w czasie czynnikach stymulujących działania strony własnej).

Do zaprojektowania systemu wykorzystano metodologię modelowania systemów „CASE METHOD". Narzędzia tej metodologii dostarczają mechanizmów umożliwiających kontrolę spójności oraz kompletności systemu. Metoda ta opiera się na przedstawianiu w postaci diagramów różnych aspektów tworzonego oprogramowania. Graficzna reprezentacja danych jest znacznie lepiej przyswajana przez człowieka niż

[24] H. Spustek, Przewaga w walce i operacji, rozprawa habilitacyjna, AON, Warszawa 2002, s. 67-80.

opisy słowne i we wstępnej fazie tworzenia systemu jest ona praktycznie jedyną formą dokumentacji czytelną dla przyszłego użytkownika.

Następnym krokiem było zaimplementowanie zaprojektowanych funkcji systemu. Do wykonania aplikacji skorzystano z oprogramowania:

- firmy Borland Inprise Corporation – DELPHI Enterprise ver. 5.0,
- firmy Microsoft Corporation – Excel 97.

ZASADNICZE CELE

Zasadniczym celem systemu informatycznego jest wspomaganie procesu określania stosunku sił stron konfliktu. Podstawą wyznaczenia strony, która na początku starcia będzie miała przewagę jest znajomość rodzaju i liczebności użytych środków walki.

Powodzenie realizacji założonego celu jest uwarunkowane spełnieniem podrzędnych celów, do których zaliczyć można:

- generowanie stosunku sił na początku starcia,
- generowanie wykresów zmian stosunku sił w czasie trwania starcia (przy stałych w czasie czynnikach stymulujących działania strony własnej),
- generowanie wykresów zmian stosunku sił w czasie trwania starcia (przy zmiennych w czasie czynnikach stymulujących działania strony własnej).

Cele te są realizowane przez funkcje opisane poniżej.

ARCHITEKTURA SYSTEMU

Architektura proponowanego systemu jest oparta na strukturalnym charakterze podejmowania decyzji przez dowódców biorących udział w konflikcie. W związku ze strukturalnym charakterem tych procesów, architektura systemu powinna być tak zaprojektowana, aby ułatwić i przyspieszyć podjęcie właściwej decyzji, dając pewne warianty rozwiązań. Niezależnie od szczebla dowodzenia znajomość stosunku sił zgrupowania wojsk własnych i przeciwnika pozwoli podając właściwą decyzję, co do rodzaju prowadzonego działania

PROPONOWANA STRUKTURA ORGANIZACYJNA SYSTEMU

Proponowany system określania przewagi jednej zez stron konfliktu jest systemem autonomicznym, mogącym funkcjonować na każdym szczeblu dowodzenia. Informacja występująca na jednym szczeblu dowodzenia nie ma bezpośredniego wpływu na informację z innego szczebla, dlatego nie ma potrzeby komunikowania się systemu pomiędzy poszczególnymi szczeblami.

ZAŁOŻENIA FUNKCJONALNE SYSTEMU

System określania przewagi stron konfliktu jest przewidziany jako narzędzie informatyczne wspomagające pracę analityków i dowódców na każdym szczeblu struktury sił zbrojnych. W związku z tym w tej części opracowania zostaną przedstawione podstawowe funkcje, jakie taki system powinien umożliwiać.

Zasadniczym zadaniem tego systemu jest umożliwienie przechowywania i odpowiedniego zarządzania danymi o środkach walki strony własnej i sił przeciwnika. Do informacji tych należy zaliczyć dane ilościowe i jakościowe charakteryzujące możliwości bojowe każdej ze stron konfliktu. Aby umożliwić wykonanie analiz stosunku sił stron konfliktu, system powinien dawać możliwość przeprowadzania aktualizacji zarówno stanów ilościowych jak i współczynników jakościowych poszczególnych środków walki.

Dokładna specyfikacja funkcji systemu została przedstawiona poniżej.

ZARZĄDZANIE DANYMI ILOŚCIOWYMI

Zarządzanie danymi ilościowymi obejmuje dwie główne funkcję: przechowywanie i aktualizację danych.

ZARZĄDZANIE DANYMI SŁOWNIKOWYMI

Danymi, które mają największy wpływ na poprawność działania systemu i rzetelność otrzymywanych wyników, są dane słownikowe. Poprawne zarządzanie tymi danymi stanowi podstawę do zastosowania algorytmów obliczeniowych w module określania stosunku sił dwóch przeciwnych stron. Dane słownikowe zostały podzielone na dwie grupy dotyczące opisu środków walki:

- słownik środków walki,
- słownik współczynników jakościowych środków walki.

Słownik środków walki.

Słownik środków walki służy do określenia jakie środki walki mogą być użyte przez strony konfliktu. Decyzja o ujęciu określonych środków walki w słowniku zależy od użytkownika. Nie ma ograniczeń formalnych co do rodzaju ujętych w słowniku środków walki, jedynym warunkiem na który należy zwrócić uwagę jest problem z późniejszym wiarygodnym przypisaniem współczynników jakościowych do tych środków walki.

Słownik współczynników jakościowych środków walki.

W celu określenia przewagi jednej ze stron konfliktu znany musi być potencjał bojowy użytych w starciu środków walki. Do tego celu służą współczynniki jakościowe określone dla każdego ze środka walki i zebrane w słowniku współczynników jakościowych środków walki. Dodatkowo dla każdego współczynnika jakościowego środka walki należy określić:

A. Czy jego wpływ na środek walki jest „stymulujący" czy „destymulujący"?
B. Jego „ważność" wobec wszystkich współczynników przypisanych dla danego środka walki.

Dane słownikowe, jak sama nazwa wskazuje są danymi stałymi i wszelkie dozwolone modyfikacje pociągają za sobą zmiany w danych pierwotnych - zgodnie z zasadami integralności bazy danych. System udostępnia dwie funkcje działające na danych słownikowych:

- przechowywanie danych,
- aktualizowanie danych.

ZARZADZANIE DANYMI PIERWOTNYMI

Dane pierwotne (rzeczywiste) służą do określenia stanów ilościowych środków walki danego zgrupowania wojsk. Służą do określenia ile danego środka walki znajduje się w posiadaniu wojsk własnych i wojsk przeciwnika. Dane te są niezbędne do późniejszych obliczeń i określania stosunku sił. System udostępnia dwie funkcje działające na danych słownikowych:

- przechowywanie danych,
- aktualizowanie danych.

ZARZADZANIA DANYMI JAKOŚCIOWYMI

Danymi, których prawdziwość ma największy wpływ na poprawność działania systemu i otrzymywane wyniki, są dane jakościowe środków walki. Pod tym pojęciem rozumiane są w systemie dane liczbowe określające wielkość wpływu danego współczynnika jakościowego na odpowiadający mu środek walki. Zakres liczbowy przypisywany określonym współczynnikom zależy od wiedzy tematycznej użytkownika i nie jest formalnie ograniczony przez system. Należy jedynie pamiętać, że część współczynników może wpływać „destymulująco" na środek walki i wtedy im mniejsza wartość jest przypisana tym dany środek w walce jest lepiej wykorzystany. Rzetelność tych danych stanowi podstawę do właściwego zastosowania algorytmów obliczeniowych i otrzymania wiarygodnych wyników. System udostępnia dwie funkcje działające na danych jakościowych:

- przechowywanie danych,
- aktualizowanie danych.

PRZETWARZANIE WYNIKOWE

Funkcja obliczania potencjału bojowego zgrupowania wojsk własnych i przeciwnika bazuje na metodzie taksonomii numerycznej opartej na algorytmie opracowanym przez Hugo Steinhausa. W pracy skorzystano z opracowanego przez autora algorytmu TAX. Algorytm ten przenosi założenia metody taksonomii numerycznej na pole działań sił zbrojnych.

ZARZADZANIA DANYMI WYNIKOWYMI

Funkcja danych wynikowych służy do graficznego zobrazowania wyników przetwarzania. System zapewnia wszelkie możliwe raporty pozwalające na pełną analizę przebiegu starcia. Dostępne są następujące funkcje:

- zobrazowanie wyników na ekranie,
- zapis wyników.

MODEL FUNKCJI W SYSTEMIE - DIAGRAMY PRZEPŁYWU DANYCH (DFD)

Diagram przepływu danych służy do przedstawienia modelu procesów w systemie. Zespół diagramów DFD dla systemu wraz z opisem elementów występujących na diagramie stanowi model procesów w systemie. Jest to graficzna „mapa" procesów ukazująca przepływ danych między procesami w systemie oraz między światem zewnętrznym a systemem. Diagram DFD buduje się z czterech podstawowych symboli: obiektów zewnętrznych, składnic albo magazynów danych, procesów i przepływów danych.

Obiekty zewnętrzne reprezentują źródła lub miejsca przeznaczenia informacji, które są zewnętrzne w stosunku do systemu. Obiekty zewnętrzne dostarczają informacji, która powoduje wykonywanie pewnych procesów w systemie, względnie odbierają informacje produkowane przez system. Jeżeli obiekt występuje na diagramie kilkukrotnie, to dla przejrzystości diagramu obiekt zewnętrzny jest powtórzony i zaznaczony ukośną kreską w prawym górnym rogu. W ten sposób unika się krzyżowania strzałek (przepływów danych) biegnących do jednego obiektu.

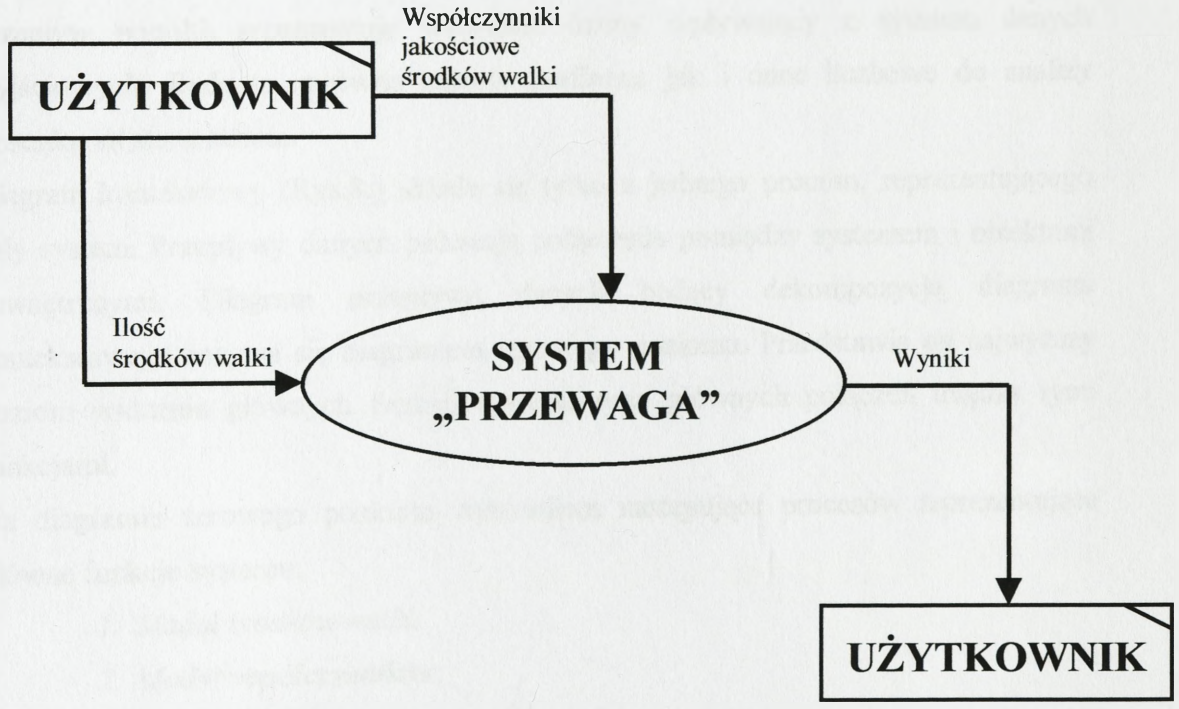
Składnice albo magazyny danych reprezentują miejsca, gdzie dane są przechowywane między procesami, które na nich operują (jeśli może wystąpić przesunięcie w czasie między tymi procesami). Magazyny te są dostępne jedynie z procesów, które mogą działać na danych w nich zawartych. Dostępność z procesów oznacza, że nie kreśli się przepływu danych bezpośrednio pomiędzy obiektem zewnętrznym a składnicą. Dla programisty lub projektanta systemów składnica zwykle przedstawia plik danych na dysku czy taśmie.

Procesy odpowiadają tym składnikom systemu, które operują na danych. Procesy otrzymują i przesyłają dane za pośrednictwem przepływów danych. Podobnie jak w przypadku składnic, proces kojarzy się z programem komputerowym lub procedurą. Przepływy danych opisują strumienie danych o określonej zawartości przepływające pomiędzy dwoma obiektami (źródłem i przeznaczeniem) na diagramie. Przedstawia się je za pomocą linii ze strzałkami określającymi kierunek przesyłania informacji. Linie są skierowane najczęściej jednostronnie. Informacja przepływa od obiektów zewnętrznych do procesów, od procesów do składnic danych, od składnic danych do procesów i od procesów do obiektów zewnętrznych.

DIAGRAM KONTEKSTOWY

Na diagramie kontekstowym przedstawiamy system, obiekty zewnętrzne (w stosunku do systemu) oddziałujące na system oraz przepływy danych łączące obiekty z systemem. Diagram kontekstowy jest szczególną postacią diagramu przepływu danych, który przedstawia podstawowe cechy systemu:

- granice systemu,
- źródła i odbiorcy informacji w systemie,
- główne wejścia i wyjścia z systemu (to znaczy informacje płynące między światem zewnętrznym a systemem).



Rys. 6. Diagram kontekstowy

Źródło: opracowanie własne

Jak widać z diagramu kontekstowego (rys. 8.) z systemem ze środowiska zewnętrznego kontaktuje się Użytkownik. Jest on obiektem zewnętrznym wobec systemu. Użytkownikiem może być np. dowództwo jednego z biorących udział w konflikcie zgrupowania wojsk.

Użytkownik zasila system następującymi informacjami: **ilość użytych środków walki, współczynniki jakościowe środków walki**. Są to przepływy danych wpływające do systemu z otoczenia. Przepływ **ilości użytych środków walki** reprezentuje dane ilościowe odnoszące się do użytych zarówno przez wojska własne, jak i przeciwnika środków walki. Przepływ **współczynniki jakościowe środków walki** reprezentuje dane opisujące właściwości bojowe środków walki.

Z systemu, do Użytkownika docierają pod postacią przepływów danych wyniki. Przepływ **wyniki,** reprezentuje wszystkie formy wpływający z systemu danych wyjściowych. Będą to zarówno raporty graficzne jak i dane liczbowe do analizy stosunku sił stron starcia.

Diagram kontekstowy (Rys.8.) składa się tylko z jednego procesu, reprezentującego cały system. Przepływy danych pokazują połączenia pomiędzy systemem i obiektami zewnętrznymi. Diagram przepływu danych będący dekompozycją diagramu kontekstowego nazywa się diagramem zerowego poziomu. Przedstawia on najwyższy poziom widzenia głównych funkcji systemu oraz głównych połączeń między tymi funkcjami.

Na diagramie zerowego poziomu wyróżniłem następujące procesów reprezentujące główne funkcje systemu:

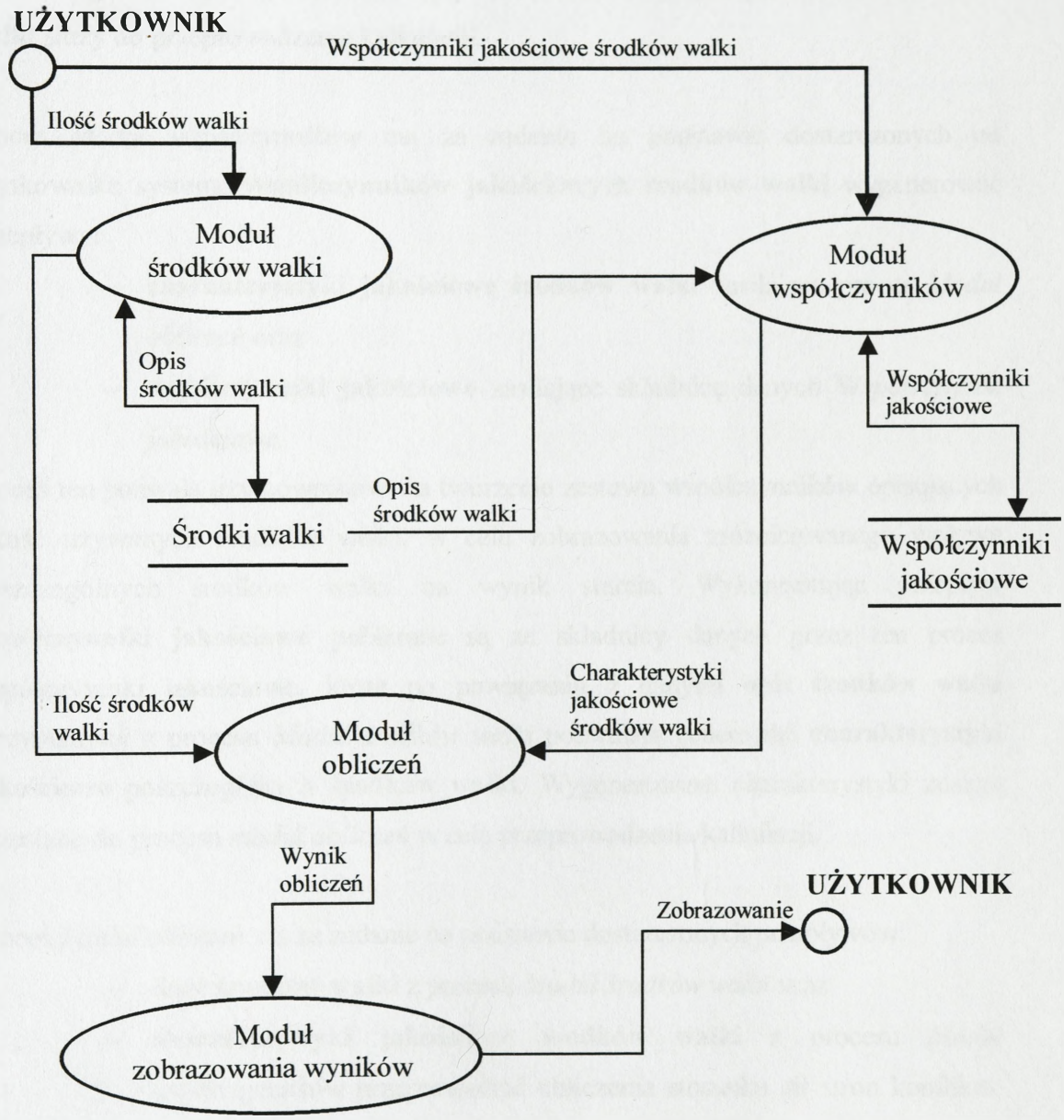
1. *Moduł środków walki;*
2. *Moduł współczynników;*
3. *Moduł obliczeń;*
4. *Moduł zobrazowania wyników.*

Proces *Moduł środków walki* ma za zadanie na podstawie dostarczonych od użytkownika systemu ilość środków walki wygenerować przepływy:

- **ilość środków walki** zasilający proces *Moduł obliczeń* oraz
- **opis środków walki** zasilający składnicę danych *Środki walki.*

Proces ten pozwala użytkownikowi na tworzenie dowolnego zestawu środków walki zależnego od założonej szczegółowości rozpatrywanego starcia.

DIAGRAM PRZEPŁYWU DANYCH POZIOMU ZEROWEGO (DFD0)



Rys.7. Diagram DFD0.

Źródło: opracowanie własne

Korzystając z **opisu środków walki** pobieranych ze składnicy danych *środki walki* proces generuje nowe **opisy środków walki**, które są ponownie zwracane do tej składnicy danych. Przepływ **ilość środków walki** przekazywany jest do procesu *moduł obliczeń*, gdzie w powiązaniu z przepływem **charakterystyki jakościowe środków walki** służy do przeprowadzenia kalkulacji.

Proces *Moduł współczynników* ma za zadanie na podstawie dostarczonych od użytkownika systemu **współczynników jakościowych środków walki** wygenerować przepływy:

- **charakterystyki jakościowe środków walki** zasilające proces *Moduł obliczeń* oraz
- **współczynniki jakościowe** zasilające składnicę danych W*spółczynniki jakościowe.*

Proces ten pozwala użytkownikowi na tworzenie zestawu współczynników opisujących jakość używanych środków walki, w celu zobrazowania zróżnicowanego wpływu poszczególnych środków walki na wynik starcia. Wykorzystując przepływ **współczynniki jakościowe** pobierane są ze składnicy danych przez ten proces współczynniki jakościowe, które po powiązaniu z danymi **opis środków walki** otrzymanymi z procesu *Moduł środków walki* pozwalają generować **charakterystyki jakościowe** poszczególnych **środków walki**. Wygenerowane charakterystyki zostają przesłane do procesu *moduł obliczeń* w celu przeprowadzenia kalkulacji.

Proces *Moduł obliczeń* ma za zadanie na podstawie dostarczonych przepływów:

- **ilość środków walki** z procesu *Moduł środków walki* oraz
- **charakterystyki jakościowe środków walki** z procesu *Moduł współczynników* przeprowadzić obliczenia stosunku sił stron konfliktu. Efektem przeprowadzonych obliczeń jest przepływ **wyniki obliczeń** wysyłany do procesu *Moduł zobrazowania wyników.*

Proces *Moduł zobrazowania wyników* ma za zadanie na podstawie dostarczonych od poszczególnych procesów przepływów danych generować przepływy do obiektu zewnętrznego, czyli do *Użytkownika.* Do procesu dochodzi tylko jeden przepływ: **wyniki obliczeń**, który zostaje przetworzony na przepływ wyjściowy **zobrazowanie.**

Na diagramie zerowego poziomu wyróżniłem następujące składnice danych:

1. *Środki walki,*
2. W*spółczynniki jakościowe.*

Składnica danych *środki walki* jest magazynem zawierającym opisy środków walki zarówno standardowych jak i nowo utworzonych. Składnica ta zasilana jest przepływem **opis środków walki** wypływającym z procesu *Moduł środków walki.* Ze składnicy dane pobierane są przez procesy *Moduł środków walki* i *Moduł współczynników.*

Składnica danych W*spółczynniki jakościowe* jest magazynem zawierającym współczynniki jakościowe środków walki. Dane z tej składnicy pobierane są przez proces *Moduł współczynników*, który jednocześnie służy do zasilenia składnicy o zaktualizowane dane.

MODEL DANYCH W SYSTEMIE - DIAGRAM ZWIAZKÓW OBIEKTÓW (ERD)

W poprzednim punkcie został przedstawiony model funkcji w systemie (ilustrowany graficznie za pomocą diagramów DFD). W tym punkcie zostanie przedstawiony sposób modelowania danych w systemie za pomocą modelu związków obiektów. Model danych służy do wyrażenia struktury danych projektowanego lub istniejącego systemu. Przez strukturę danych rozumie się typy danych występujących w systemie, wzajemne powiązania między tymi danymi oraz pewne ograniczenia nałożone na te dane. Graficzną ilustracją takiego modelu danych jest diagram związków obiektów ERD (ang. Entity Relationship Diagram), który służy do graficznego przedstawienia struktury danych systemu. Podstawowymi pojęciami modelu związku obiektów są: obiekt, związek, atrybut.

„Obiekt jest to coś, co istnieje, ma znaczenie i jest odróżnialne, o którym informacje powinny być znane lub przechowywane.[25]

[25] R. Barker: CASE METHOD: Modelowanie związków encji. WNT, Warszawa 1996.

Obiekty mogą być rzeczywiste, mogą też być pewnymi pojęciami abstrakcyjnymi. Każdy obiekt musi być jednoznacznie identyfikowalny, aby każde wystąpienie obiektu można było odróżnić od wszystkich innych wystąpień tego typu obiektów. Jednoznacznym identyfikatorem może być atrybut, kombinacja atrybutów, kombinacja związków lub kombinacja atrybutów i związków.

Atrybut jest zespołem znaczących cech, tzn. dowolnych szczegółów służących do klasyfikacji, rozróżnienia, kwalifikacji, identyfikacji lub wyrażających stan obiektu. Zatem, atrybut jest funkcją przypisującą obiektowi wartość cechy ze zbioru wartości tej cechy. Każdy obiekt musi być jednoznacznie zidentyfikowany przez kombinację atrybutów i/lub związków. Trzeba, więc tak dobierać atrybuty, aby pomogły jednoznacznie zidentyfikować obiekt. Na diagramie związków obiektów ERD główny jednoznaczny identyfikator wskazuje się, umieszczając znak „#" przed każdym atrybutem wchodzącym w skład jednoznacznego identyfikatora i stawiając poprzeczną kreskę na linii każdego związku wchodzącego w skład tego identyfikatora. W przypadku, gdy wartość atrybutu jest określona przez pewien czas lub może nie być osiągalna to stawiamy małe „o" przed nazwą atrybutu, wskazując w ten sposób jego opcjonalność. Gdy wartość atrybutu musi być zawsze określona, stawiamy mały znak „*" przed nazwą atrybutu.

Związek jest nazwanym, istotnym powiązaniem między istniejącym dwoma obiektami[26]. W szczególnym przypadku może być powiązaniem tego samego obiektu ze sobą. Każdy związek ma dwa końce, z których każdy ma przypisaną:

- nazwę,
- stopień / liczebność,
- opcjonalność.

Związek jest reprezentowany za pomocą linii łączącej dwie ramki obiektów ze sobą lub rekurencyjnie jedną ramkę obiektu ze sobą. Związek, którego liczebność wynosi „wiele" rysuje się jako linię rozgałęziającą się na trzy i dotykającą ramki obiektu, tzw. „kurze stopki". Przy liczebności „jeden" linia związku dotyka ramki w jednym punkcie. Gdy koniec związku jest wymagany, wtedy odpowiadająca mu połowa linii jest narysowana jako ciągła. Gdy koniec związku jest opcjonalny, rysujemy linię przerywaną. Nazwa każdego końca związku umieszczona jest przy każdym końcu.

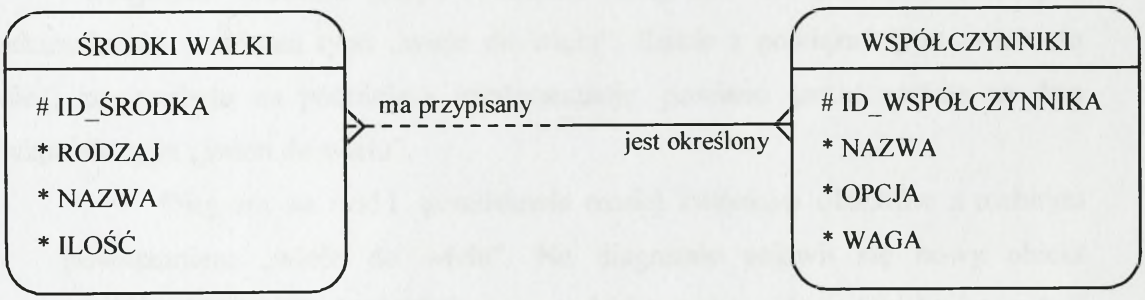
---

[26] R. Barker: CASE METHOD: Modelowanie związków encji. WNT, Warszawa 1996.

Diagram DFD przedstawia funkcjonalny model systemu, odpowiada więc na podstawowe pytanie, co system ma robić. Model danych w systemie przedstawiony za pomocą ERD służy do zilustrowania statycznej struktury danych w systemie w sposób niezależny od konkretnego systemu zarządzania tymi danymi. Model danych systemu zbudowany na podstawie modelu związków obiektów reprezentuje wszelkie obiekty oraz powiązania między nimi, istotne z punktu widzenia funkcjonowania systemu.

Diagram ERD

Diagram ten przedstawia ERD dla systemu, którego model funkcjonalny przedstawiłem na diagramie DFD (rys.8).



Rys.8. Diagram ERD

Źródło: opracowanie własne

W systemie wyróżniłem następujące obiekty:

1. ŚRODKI WALKI
2. WSPÓŁCZYNNIKI

Obiekt ŚRODKI WALKI określa zbiór wszystkich dostępnych w systemie środków walki. Identyfikatorem tego obiektu jest atrybut ID_ŚRODKA, który określa jednoznacznie dane wystąpienie obiektu (może to być na przykład indeks UiSW zgodny z tabel a indeksową używaną w SZ). Jak widać na diagramie obiekt ŚRODKI WALKI powiązany jest z obiektem WSPÓŁCZYNNIKI. Związek występujący w tym powiązaniu jest obligatoryjny o liczebności „wiele do wielu".

Obiekt WSPÓŁCZYNNIKI określa zbiór wszystkich dostępnych współczynników jakościowych służących do opisu środków walki np. zasięg, pułap,

szybkostrzelność itp. Identyfikatorem tego obiektu jest atrybut ID_WSPÓŁCZYNNIKA, który określa jednoznacznie dane wystąpienie tego obiektu. Obiekt ten powiązany jest z obiektem ŚRODKI WALKI za pomocą związku opcjonalnego na końcu o liczebności „wiele do wielu".

Diagram z rys 10. należy czytać w następujący sposób:

- od lewej do prawej

  Każdy ŚRODEK WALKI *musi* **mieć przypisany** *jeden lub więcej* WSPÓŁCZYNNIKÓW.

- od prawej do lewej

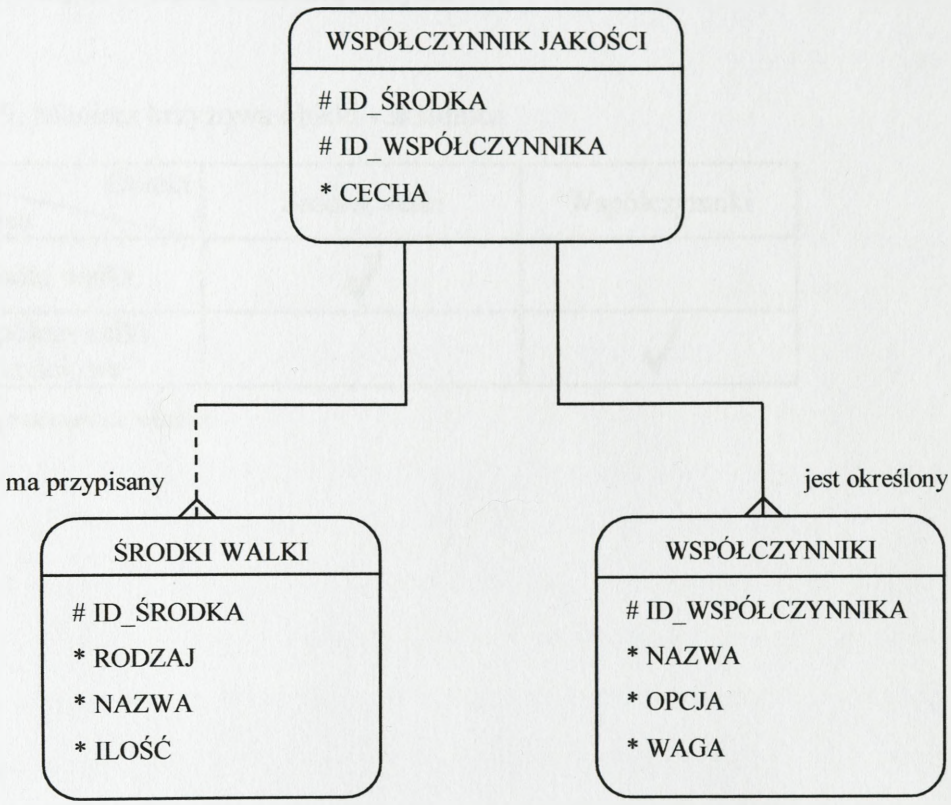
  Każdy WSPÓŁCZYNNIK *może* **być określony dla** *jednego lub więcej* ŚRODKÓW WALKI.

Następnym krokiem przy tworzeniu diagramu związku obiektów jest przekształcenie powiązań typu „wiele do wielu". Każde z powiązań typu „wiele do wielu", ze względu na późniejszą implementację, powinno zostać rozbite na dwa powiązania typu „jeden do wielu".

Diagram na rys11. przedstawia model związków obiektów z rozbitym powiązaniem „wiele do wielu". Na diagramie pojawił się nowy obiekt WSPÓŁCZYNNIK JAKOŚCI zwany obiektem intersekcji. Obiekt intersekcji (przecięcia) to obiekt powstały w wyniku eliminacji związku „wiele do wielu" między dwoma obiektami. Instancje tak powstałego obiektu intersekcji mogą istnieć tylko w kontekście dwóch określających go obiektów referencyjnych. Pod pojęciem obiektu referencyjnego (odniesienia) rozumiemy obiekt, który nie jest połączony „wymaganym" końcem żadnego związku z innym obiektem. Wprowadzone pośrednie obiekty (intersekcje) mają nazwy określające w jakim celu je zastosowano zaś ich identyfikatory stanowią identyfikatory obiektów, które ze sobą łączą.

DIAGRAM ERD - Z ROZBITYMI POWIAZANIAMI

Diagram ten przedstawia ERD z rys 10. z rozbitymi powiązaniami typu „wiele do wielu".



Rys. 9. Diagram ERD z rozbitym powiązaniem

Źródło: opracowanie własne

WERYFIKACJA DIAGRAMU DFD WZGLEDEM DIAGRAMU ERD

Weryfikacja ta jest wymagana w różnym stopniu w różnych metodykach. Można jednak powiedzieć, że otrzymane za pomocą dwóch omówionych wcześniej przeze mnie technik modelowania systemu, modele: funkcji w systemie (diagram DFD) oraz danych (diagram ERD), jako tworzące dwie wzajemnie się uzupełniające płaszczyzny widzenia systemu, muszą być ze sobą powiązane w określony sposób.

Weryfikacja tych dwóch modeli polega na dokonaniu powiązania między diagramami ERD oraz DFD utworzonymi dla danego systemu. Każdy obiekt

z diagramu ERD powinien znaleźć się w jednej lub wielu składnicach z diagramu DFD. Bardzo często przedstawia się to w postaci tablicy krzyżowej obiekt/składnica danych. W większości wypadków istnieje odpowiedniość jeden do jednego pomiędzy składnicami a obiektami. Jednakże bardzo często kilka obiektów będzie odwzorowanych w jedną składnicę danych.

Tablica 9. Macierz krzyżowa obiekt - składnica

Obiekt / Składnica	Środki walki	Współczynniki
Środki walki	✓	
Współczynniki jakościowe		✓

Źródło: opracowanie własne

ZAKOŃCZENIE

Przedstawione opracowanie zawiera materiał dwojakiego rodzaju. Po pierwsze jest to zbiór wiadomości teoretycznych dotyczących współczesnego, ciągle zmieniającego się obrazu pola walki, w tym szczególnie roli informacji, jej wpływu na wynik walki w każdym jej wymiarze. Po drugie znajdujemy tu czysto praktyczne rady, w jaki sposób wykorzystać potencjalne możliwości tkwiące w prostych narzędziach informatycznych. Narzędziami tymi nazwano arkusz kalkulacyjny Excel z jego zaawansowanymi dodatkami analitycznymi (SOLVER) oraz środowisko programistyczne DELPHI. Dużą porcję informacji praktycznych, mogących się przydać informatykom-praktykom zawiera rozdział trzeci. Na bazie zgromadzonej tam wiedzy, autor opracowania wykonał własne narzędzie obliczeniowe. Narzędzie to jest implementacją numeryczną modelu przewagi będącego w całości osiągnięciem autora niniejszego opracowania.

W dotychczasowych, publikacjach dotyczących modelowania zjawisk pola walki stosowano dwojakiego rodzaju podejście metodologiczne: deterministyczne, oparte na modelu walki Lanchestera oraz stochastyczne, będące wynikiem randomizacji deterministycznych liniowych równań różniczkowych. Pierwsze podejście, nie przystaje do współczesnego pola walki, zaś drugie – mocno skomplikowane matematycznie, daje wprawdzie wiarygodne wyniki lecz wymaga wielu zmiennych wejściowych, dużej ilości różnego rodzaju współczynników, przeważnie empirycznych, co sprawia że ich praktyczne użycie staje się kłopotliwe. Model przewagi, stworzony przez autora niniejszego opracowania a szczegółowo opisany w jego rozprawie habilitacyjnej, do której autor odnosi się w niniejszym opracowaniu, stanowi unikalne podejście do modelowania zjawisk współczesnego pola walki, czego nie było w dotychczasowych, fragmentarycznych opracowaniach dotyczących tego typu problemów. Model Joshua-Epsteina (opracowany na potrzeby analiz danych historycznych z konfliktów mających miejsce współcześnie) został przez autora odrzucony. Model ten działa na kilkunastu współczynnikach zadawanych na wejściu. Wymaga to znajomości szczegółowych danych o zgrupowaniach wojsk biorących udział w konflikcie. Tymczasem, tak

precyzyjnych informacji w czasie trwania konfliktu zwykle nie posiadamy. Natomiast, model stworzony przez autora dysertacji jest wolny od tych wad bowiem u podstaw modelu leżą dwa zasadnicze założenia: model ma generować wartości przybliżone (liczą się jedynie prognozy na kilka najbliższych przedziałów czasowych), strony uczestniczące w konflikcie zbrojnym pozostają w izolacji w bliskim horyzoncie czasowym. Wychodząc z liniowego równania różniczkowego określającego prawo wzrostu Robertsona, po odpowiednich przekształceniach algebraicznych, otrzymuje się wymaganą postać krzywej logistycznej, stanowiącej punkt wyjścia do opracowania oryginalnego, dynamicznego modelu przewagi. Przedstawiony model stanowi ilościowe powiązanie pojęć: „potencjał bojowy", „przewaga" oraz „stosunek sił".

W dynamicznym modelu przewagi, miarą przewagi początkowej jest wyjściowy stosunek sił stron pozostających w konflikcie zbrojnym. Autor opracował całkowicie nową metodę wyznaczania potencjałów bojowych zgrupowań wojsk opartą na wielokryterialnej analizie porównawczej. Stosowana dotychczas metoda wyznaczania potencjałów bojowych wojsk bazuje na stałych w czasie współczynnikach jakości taktyczno-technicznej, nie uwzględnia ani systemu wag, ani też rodzaju oddziaływań na które autor zwrócił uwagę. Zaproponowana metoda obliczania potencjału bojowego zgrupowań wojsk jest wolna od tych wad i generuje różne wyniki, zależnie od wag nadanych poszczególnym oddziaływaniom w odróżnieniu do metody stosowanej obecnie, która często daje mylne wyniki. Stworzony model przewagi został przetestowany na literaturowych danych empirycznych. Weryfikacja modelu została przeprowadzona poprzez porównanie wyników generowanych przez model z wynikami uzyskanymi przy użyciu modelu Joshua – Epsteina i dała pozytywny wynik, co potwierdziło poprawność modelu.

Dalsze prace nad modelem przewagi i jego implementacją numeryczną będą kontynuowane w przyszłości.

LITERATURA

1. R. Barker, CASE METHOD: Modelowanie związków encji, WNT, Warszawa 1996.
2. R. Barker, C. Longman, CASE METHOD: Modelowanie funkcji i procesów, WNT, Warszawa 1996.
3. R. Darilek i inni, Miary efektywności dla armii wieku informacyjnego, Centrum Rand Arroyo, 2002 – tłumaczenie z języka angielskiego, źrdło: http//www.rand.org/organization/ard/.
4. Dieter W.Heermann, Podstawy symulacji komputerowych w fizyce, WNT Warszawa 1997.
5. E. Gatnar, Symboliczne metody klasyfikacji danych PWN, Warszawa 1998.
6. H.Hotelling, Differential Equations Subject to Errors and Population Estimates, Journal of the American Statistical Association, 1927.
7. M. Huzarski, Taktyka ogólna wojsk lądowych, AON, Warszawa 2001.
8. A.Janicki, A.Izydorczyk, Komputerowe metody w modelowaniu stochastycznym, WNT, Warszawa 200.
9. W. Kaczmarek, Brygada zmechanizowana (pancerna) w obronie i natarciu, Warszawa 1995.
10. A.M.Matwiejew, Fizyka cząsteczkowa, PWN, Warszawa 1989.
11. J.Michniak, Metody i treści pracy zespołów funkcjonalnych na SD WLąd, AON Warszawa 2000.
12. E. Nowak, Metody taksonomiczne w klasyfikacji obiektów społeczno-gospodarczych, PWE, Warszawa 1990.
13. Podstawowe kalkulacje operacyjno-taktyczne, MON SG WP, Warszawa 1988.
14. S. Smolik, Wyznaczanie parametrów krzywej logistycznej, Przegląd statystyczny R.XXXII- zeszyt 4-1985.
15. H. Spustek, K. Loch, Wielokryterialna analiza porównawcza, problem n.b. 12/7.4, AON, Warszawa 1995.
16. H. Spustek, Analizy porównawcze potencjałów bojowych zgrupowań wojsk w konflikcie zbrojnym, AON, Warszawa 2002.
17. H. Spustek, Przewaga w walce i operacji, rozprawa habilitacyjna, AON, Warszawa 2002.
18. M. Sułek, Podstawy potęgonomii i potęgometrii, Kieleckie Towarzystwo Edukacji Ekonomicznej, Kielce 2001.
19. J. Wołejszo, Sposoby obliczania potencjału bojowego pododdziału, oddziału i związku taktycznego, AON, Warszawa 2000.
20. K.Zalewski, Wykłady z termodynamiki fenomenologicznej i statystycznej, PWN Warszawa 1978.

ZAŁĄCZNIK

DYNAMIC SUPREMACY MODEL APPLIED TO MILITARY

Summary

The chance to fulfil a task on a contemporary battlefield (the probability to succeed) is tightly connected to the problem of defining potential abilities of own forces/armies in relation to the forces of the enemy.

There are various theoretical methods of measuring combat potential of parties participating in the military conflict. However, all of those methods have one serious disadvantage. Most often, their authors use a great number of various factors to describe phenomena happening on a battlefield. Those factors are of empirical character, particularly, in relation to evaluation of the armament possessed by both parties. The above approach leads to some difficulties of calculation nature. Sometimes you are simply not able to use a given method because of the lack of sufficient data.

The following assumptions constitute the base of a suggested model of supremacy determination:

1. The ratio between the parties' forces – defined as 'supremacy' – has a direct impact on the combat potential of the military formations.
2. The proportion of forces is influenced by not measurable factors.
3. The above phenomenon may be described in an analytical way.
4. Modelling of the supremacy process is possible.

The article presents a simple, analytical model, describing the phenomenon of creating supremacy on a contemporary battlefield. Theoretical considerations are based on a logistic curve . Thus, we were able to achieve a clear, physical interpretation of phenomena described, and at the same time, to reduce the number of parameters necessary to construct the model, to two quasi constant α and β factors.

The purpose of this project was to build a simple, in a mathematical sense, model that would enable measurement of chances to perform an assigned task under the conditions of activity that are precisely defined.

Simultaneously, the model has a simple construction that enables its easy implementation, in a form of numeric programme supporting the decision making process on a contemporary battlefield.

1. Introduction

Multicriterion comparative analysis is a tool supporting decision making process while choosing a technical system. Such analysis may be conducted using various research techniques. Analyses that lead to a choice or determination whether defined armament systems are useful may be defined as static analyses of technique evaluation. Evaluation of techniques (of armament in particular) is a fundamental issue and constitutes a starting point to further, much more sophisticated analyses related to determination and comparison of combat potential of military formations. However, such analyses should not be of static character.

In this article we tried to deal with a difficult problem of evaluating combat potentials of parties participating in a military conflict. The problem is considered as dynamic, and being in a tight relation to the question of supremacy creation. Measurable consideration of influence of immaterial factors on supremacy creation phenomenon constitutes additional problem.
Analysis of factors influencing the quantity called 'supremacy' evidently suggests the need to separate two categories:

- factors that stimulate supremacy (leading to the increase of supremacy) represented by factor α,
- factors decreasing the value of supremacy, represented by factor β.

Both types of factors: α and β are formed of measurable (quantitative) factors and not measurable, or hard measurable (qualitative) factors. Increase in supremacy over the enemy may be unequivocally defined through calculation of proportion of losses of fighting parties throughout successive days of conflict. Proportion of losses of parties in conflict is variable in time and its value is reflected by the influence of α and β factors during the battle course. Credibility of the achieved results is dependent of proper definition of these factors.

2. Proposition of Mathematical Model

Initial forces of the two parties R and B are given; their value shall be defined as R_0 and B_0. The notion 'Initial forces' does not stand only for strength/quantity of armies but shall be understood much broadly. Initial forces determine potential capabilities of parties participating in the conflict. They are major determinants of the success or loss of future military operations. Thus, the abovementioned initial forces constitute values of the entire initial combat potential of parties being in a military conflict. Therefore, I suggest the proportion of forces should have the following form:

$$\frac{R}{B}(t) = A \cdot f[\alpha(t), \beta(t)], \qquad (1)$$

where, $\alpha(t)$ and $\beta(t)$ are accordingly factors that stimulate and hinder operations of the party R In relation to the party B (stimulate and hindering factors),
A - is constant,
t - time.

The following relation takes place between factors α and β:

$$\alpha(t) + \beta(t) = 1, \text{ for } t = 1,2,3,\ldots,t_{\max}, \qquad (2)$$

(In such situation we talk of quasi constant factors α and β).

Depending on value $\frac{R}{B}(t)$ In a given moment of time, we can talk of the supremacy over the enemy (value above the unit) or lack of such supremacy. Value of constant A is easy to calculate (on condition the form of function $f[\alpha(t), \beta(t)]$ of the following relation is given:

$$\frac{R}{B}(t_0) = \frac{R_0}{B_0} = A \cdot f[\alpha(t), \beta(t)],$$

$$A = \frac{R_0}{B_0 \cdot f[\alpha(t_0), \beta(t_0)]}. \qquad (3)$$

The key problem is a definition of the form of function $f[\alpha(t), \beta(t)]$, in a way the process of supremacy creation may be best described.

The function should take the value equal the unit for α=1 and β=0 (the enemy is passive/ surprised). In case α=0 and β=1, the value of function should be undefined – absurd initial conditions, the fight has no sense if the stimulate factor equals zero.
Considering the form of relation determining the proportion of forces of the fighting parties as (1) and considering the condition (3) assuming invariability of factors α and β during the initial stage of the battle, I suggest the relation of forces of the fighting parties should be expressed in the following way:

$$\frac{R}{B}(t) = \begin{cases} \frac{R_0}{B_0} & dla \quad \alpha = 1, \beta = 0 \\ \frac{e^{\alpha t}}{e^{\alpha t} - 1 + k'} & dla \quad \alpha \in (0,1) \\ nieokreslony & dla \quad \alpha = 0, \beta = 1 \end{cases}, \qquad (4)$$

where: $k' = \left(\frac{R_0}{B_0}\right)^{-1}$ stands for the initial proportion of combat potential participating in the conflict.

3. Justification

Beneath, you can find justification of the above dependence:

I. Values R_0 and B_0 defined as the initial forces of the parties are considered in relation to qualitative and quantitative parameters. Values R_0 and B_0 are influenced by two groups of factors: material and non material. The group of non material factors comprise, among others, the level of training of armies and commands, the level of combat readiness, discipline, abilities to use broadly understood battle environment, organisational structure – the level of its functionality.

II. Factors α and β reflect mechanisms having impact, either positive or negative, on the result of battle and, at the same time the *supremacy over the enemy*. In order to define numerical values of those factors we need to know the essence of supremacy creation and define measurable way of its evaluation. We should also consider the fact that the mentioned factors do not have to and, certainly, they are not constant values. We should also reckon that the reliable analysis of the phenomenon of gaining the supremacy shall demand variability of factors α and β. The dependence (4) should be then modified in a way that constant values of the above factors should be replaced by factors dependent of the course of battle within the period preceding the moment *t*, that is, at the moment $t-1$. Then, the relation (4) shall take the following form:

$$\frac{R}{B}(t) = \begin{cases} \frac{R_0}{B_0} & dla \quad \alpha(t_0) = 1, \beta(t_0) = 0 \\ \frac{e^{\alpha[t-1]}}{e^{\alpha[t-1]} - 1 + k'[t-1]} & dla \quad \alpha(t) \in (0,1) \\ nieokreslony & dla \quad \alpha(t) = 0, \beta(t) = 1 \end{cases} \qquad (5)$$

The relation (1) was introduced as modification of the solution of known differential equations constituting the model of Lanchester battle. English Mathematician F. W. Manchester, defined

several simple and incredibly interesting battle models. Equations presented in Lanchester Model determine course of the battle in an analytical way. Basing on these equations we can calculate the number of soldiers and the amount of losses for a given moment of time.

The equations succeeded while analysing battles of antique wars and the period of firearms. They were also used in sophisticated operations in a general and broader form. Unfortunately, each alteration – modification of the equations was related to the increase in the number of variables and subsequent factors added to the model. On the other hand, the classic Lanchester model may, in no way, be used directly to solve the problems of contemporary field of military operations. Therefore, we need to find the indirect solution – the model combining some aggregated factors, that simultaneously would be simple in an analytical sense.

Lenchester equations have the following form:

$$\begin{cases} \frac{dR(t)}{dt} = -bB(t)^{c_1} R(t)^{c_2} \\ \frac{dB(t)}{dt} = -rR(t)^{c_3} R(t)^{c_4} \end{cases}, \qquad (6)$$

where: r and b are real numbers from the range (0, 1), so called Lanchester efficiency factors.

Those factors are sometimes interpreted as the probability of loss of each fighting party within one day of battle (daily loss probability),
$c_1, \ldots, c_4$ indices of powers, where B(t) and R(t) may take numeric values of 0 or 1.

Derivatives $\frac{dR(t)}{dt}$ and $\frac{dB(t)}{dt}$ stand for, so called, lethality (losses) of the parties.

Having conducted proper mathematical transformation we gain four kinds of solutions, depending on the values of factors $c_1, \ldots, c_4$ (see table 1).

Solution type	c_1	c_2	c_3	c_4	Solution
quadratic	1	0	1	0	$b[B(0)^2 - B(t)^2] = r[R(0)^2 - R(t)^2]$
linear	1	1	1	1	$b[B(0) - B(t)] = r[R(0) - R(t)]$
logarythmic	0	1	0	1	$b\ln[B(0)/B(t)] = r\ln[R(0)/R(t)]$
ambush	1	1	1	0	$b/2[B(0)^2-B(t)^2] = r[R(0) - R(t)]$

Table 1. Possible versions of solution of the system of Lanchester equations.[1]

Let's see the version of quadratic equation. In this case Lanchester equations take the following form:

$$\begin{cases} \frac{dR(t)}{dt} = -bB(t) \\ \frac{dB(t)}{dt} = -rR(t) \end{cases}. \qquad (7)$$

Suggesting the following solution:

[1]You will find the precise analysis of considerations presented above in the work of Joshua M. Epstain, The Calculus of Conventional War Dynamic Analysis without Lanchester Theory, Studies in Defense Policy

$$B(t) = Ae^{\alpha t} + Be^{-\alpha t}$$

and using Table 1 in order to define the form of solution $R(t)$ we have:

$$B(t) = \frac{1}{2}\left\{\left[B(0) - \sqrt{r/b}R(0)\right]e^{\sqrt{rb}t} + \left[B(0) + \sqrt{r/b}R(0)\right]e^{-\sqrt{rb}t}\right\} \quad (8)$$

and

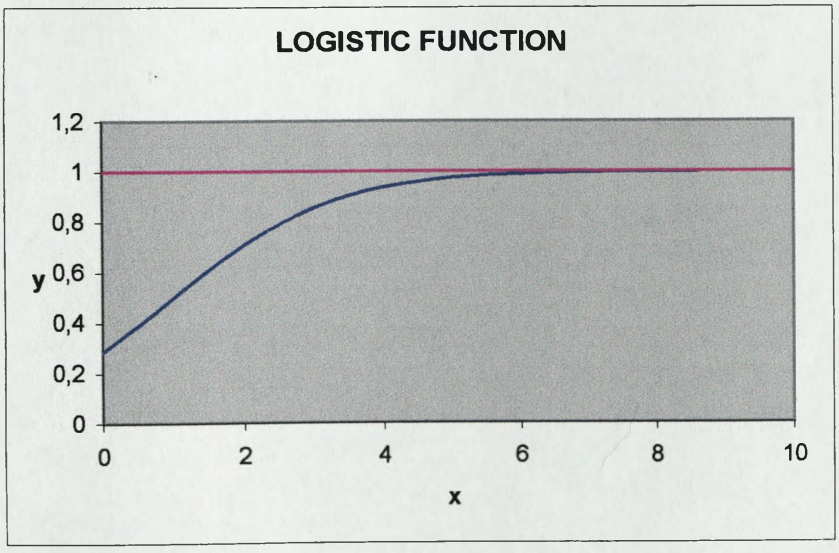
$$R(t) = \frac{1}{2}\left\{\left[B(0) - \sqrt{b/r}B(0)\right]e^{\sqrt{rb}t} + \left[R(0) + \sqrt{b/r}B(0)\right]e^{-\sqrt{rb}t}\right\}. \quad (9)$$

The above solution was achieved assuming, factors of operational efficiency r and b are constant in time – compare to the equity (1). Constant factors r and b were replaced with α and β factors variable in time. To calculate the time of battle you have only to assume the equity (9) equals zero and you have to count time t_k. Making proper transformations we get the following:

$$t_k = \left(\frac{1}{rb}\ln\frac{R(0) + \sqrt{b/r}B(0)}{\sqrt{b/r}B(0) - B(0)}\right)^{1/2}. \quad (10)$$

Basing on the relation (10) we notice that the time of battle depends exclusively on constant factors r and b and the initial strength of the parties, which is absolutely unacceptable.

II. The proposed dependence defining, variable in time, proportion of fighting parties was established basing on logistic function analysis[2].



$$f(x) = \frac{a}{1 + b \cdot e^{-cx}}$$

$a = 1$
$b = 2{,}5$
$c = 0{,}9$

Logistic function occupies a privileged position among various forms of functions used to describe economical and environmental phenomena. This results at the fact that both growth of living organisms and their amount, as well as many social and economical phenomena may be described using this curve. The equity of the above curve is gained in course of deduction coming from physical considerations. With the logistic function equation you may describe all phenomena, whose speed of growth is directly proportional to the multiplication product of the achieved growth, defined as momentum ratio and, to the distance of this growth from the level

[2] S.Smolik Przegląd statystyczny R. XXXII – volume 4 - 1985

of saturation $(a-y)$, which is called braking ratio (since value of this ratio decreases together with the flow of time). We gain differential equation (known as Robertson's Law):

$$\frac{dy}{dx} = ky(a-y), \qquad (11)$$

where, $k > 0$ is a proportion ratio,
a is the level of saturation (Y-axis asymptote of a logistic curve).

Having integrated we get:

$$\left(\frac{1}{y} + \frac{1}{a-y}\right)dy = akdx, \qquad (12)$$

and further $$\frac{y}{a-y} = e^{akx+C_1}, \qquad (13)$$

where C_1 optional constant.

Thus $$y = \frac{a}{1+b\cdot e^{-akx-C_1}}. \qquad (14)$$

Assuming $c = ak$, $b = e^{-C_1}$, we get the following equal form of the logistic curve:

$$y = \frac{a}{1+be^{-cx}}. \qquad (15)$$

This form of curve was used in this article. The notion of parameter a is a simple result of the character of supremacy phenomenon.
The breach of fight takes place when $a = 1$, which corresponds to even combat potentials of the parties R and B. Factor *k* defining the speed of saturation phenomenon development (that is: the speed at which the forces – combat potentials of fighting parties - become equal) is equal to the hindering (unfavourable) ratio β. Finally, we have to count ratio b presented in the formula (15). This is not complicated, we need only to recall the method of calculating the constant A - see (3).

$$\begin{aligned} &t = 0 \Rightarrow \frac{R}{B}(0) = \frac{R_0}{B_0} = \frac{1}{1+be^0} \\ &R_0(1+b) = B_0 \\ &b = \frac{B_0}{R_0} - 1 \end{aligned} \qquad (16)$$

Now, we need only to place the dependence (16) at (15) and make elementary algebraic transformations to get the dependence (4).

$$\begin{aligned} \frac{R}{B}(t) &= \frac{1}{1+\left(\frac{B_0}{R_0}-1\right)\cdot e^{-\beta t}} = \frac{R_0 e^{\beta t}}{R_0\left(e^{\beta t}-1\right)+B_0} = \\ &= \frac{B_0 \cdot \frac{R_0}{B_0} e^{\beta t}}{B_0 \cdot \left[\frac{R_0}{B_0}\left(e^{\beta t}-1\right)+1\right]} = \frac{k' \cdot e^{\beta t}}{k' \cdot \left(e^{\beta t}-1\right)+1} = \\ &= \frac{e^{\beta t}}{e^{\beta t}-1+\left(k'\right)^{-1}} \end{aligned} \qquad (17)$$

Let's now verify the condition (3):

$$t_0 = 0 \quad \Rightarrow \quad \frac{R}{B}(0) = k' = \frac{R_0}{B_0}. \tag{18}$$

4. Conclusions

- ✓ Model correctness was verified through establishing the degree of accordance of the achieved results with the results generated by Epstein's model[3]. The calculations were conducted basing on the following assumptions: R_0 and B_0 stand for initial potentials of the parties, α and β are constant throughout the whole period of battle. For α and β, constant in time, we gained a satisfactory accordance of results with Epstein's model at the level of a per cent, relative error of 1% for the initial stage of the battle. In further time range this error significantly increases up to 27%. Therefore, we may state that the model may be used to describe the initial/first stage of battle. Furthermore, the level of model complexity is much lower that Epstein's, which results at easier and shorter calculations.
- ✓ Since the construction of the model is very simple, it is possible to promptly create computer application supporting decision making process on the contemporary battlefield.
- ✓ Further modifications of the model, through making the factors α and β variable in time, are also possible.

BIBLIOTEKA GŁÓWNA - ARCHIWUM
Nr ewid. 68614
Akademii Obrony Narodowej

[3] Joshua M. Epstain, The Calculus of Conventional War Dynamic Analysis without Lanchester Theory, Studies in Defense Policy