UNIWERSYTET IM. ADAMA MICKIEWICZA
WYDZIAŁ MATEMATYKI I INFORMATYKI

Jacek Michałowski
nr albumu: 294063

Statystyczna weryfikacja hipotez dotyczących konwencji brydżowych

Statistical verification of hipotheses concerning bridge conventions

Praca magisterska na kierunku:
INFORMATYKA
Specjalność:
OGÓLNA
Promotor:
prof. Krzysztof Jassem

Poznań 2016

Poznań, dnia 21.06.2016

OŚWIADCZENIE

Ja, niżej podpisany Jacek Michałowski, student Wydziału Matematyki i Informatyki Uniwersytetu im. Adama Mickiewicza w Poznaniu, oświadczam, że przedkładaną pracę dyplomową pt. *Statystyczna weryfikacja hipotez dotyczących konwencji brydżowych* napisałem samodzielnie. Oznacza to, że przy pisaniu pracy, poza niezbędnymi konsultacjami, nie korzystałem z pomocy innych osób, a w szczególności nie zlecałem opracowania lub jej części innym osobom, ani nie odpisywałem tej rozprawy lub jej części od innych osób.

Oświadczam również, że egzemplarz pracy dyplomowej w wersji drukowanej jest całkowicie zgodny z egzemplarzem pracy dyplomowej w wersji elektronicznej.

Jednocześnie przyjmuję do wiadomości, że przypisanie sobie, w pracy dyplomowej, autorstwa istotnego fragmentu lub innych elementów cudzego utworu lub ustalenia naukowego stanowi podstawę stwierdzenia nieważności postępowania w sprawie nadania tytułu zawodowego.

Wyrażam zgodę na udostępnienie mojej pracy w czytelni Archiwum UAM oraz na udostępnienie mojej pracy w zakresie koniecznym do ochrony mojego prawa do autorstwa lub praw osób trzecich.

…….

(podpis)

STRESZCZENIE

Wraz z rozwojem brydża (zwłaszcza sportowego), coraz bardziej stawało się oczywiste, że przebieg licytacji ma poważny wpływ na rezultat rozdania. Z tego powodu powstało wiele konwencji i systemów licytacyjnych. Każda konwencja licytacyjna oparta jest na pewnych założeniach, które potocznie określane są mianem "praw brydżowych". Stanowią one pewną analogię dla twierdzeń matematycznych, jednak nawet te, które są od dawna uważane przez społeczność brydżystów za prawdziwe, na ogół oparte są raczej na doświadczeniu, obserwacjach, czy też intuicjach brydżystów, a nie na fundamentach naukowych.

Celem niniejszej pracy jest sprawdzenie, czy te "prawa brydżowe" (traktowane jako hipotezy) można weryfikować w oparciu o badania statystyczne. Pozwoliłoby to na zbudowanie naukowych podstaw, które następnie mogłyby zostać wykorzystane do weryfikacji konwencji już istniejących oraz konstruowania nowych. Poza testowaniem znanych hipotez, postawione zostaną też kolejne, a także zbadane zostaną zależności między nimi. Opisane będą również statystyczne metody rozwiązywania problemów decyzyjnych (w licytacji), oraz nowa metoda weryfikacji losowości danego zbioru rozdań. Na potrzeby prowadzonych badań powstał projekt, na który składają się: generator rozdań z narzuconymi ograniczeniami, program do rozwiązywania problemu rozgrywki w otwarte karty (autorstwa Piotra Belinga), oraz narzędzia do obliczeń statystycznych.

Praca składa się z czterech części. Pierwszą z nich stanowią rozdziały 2. (w którym zostały zdefiniowane podstawowe pojęcia wykorzystane w pracy) i 3. (w którym przedstawiony został obecny stan prac i już istniejące podobne rozwiązania). Drugą część stanowi podrozdział 4.1., gdzie opisane zostały rozwiązania zastosowane w projekcie magisterskim. W części trzeciej (4.2.) zostały opisane przeprowadzone eksperymenty, z których pierwszych sześć polegało na weryfikacji hipotez ("praw brydżowych"), natomiast pozostałe trzy stanowiły przykłady rozwiązywania problemów decyzyjnych w licytacji. W ostatniej części (rozdział 5.) przedstawione zostały rezultaty eksperymentów.

ABSTRACT

Along with the development of bridge (especially duplicate bridge), it became more and more obvious that the course of the auction has a serious impact on board result. That’s why many conventions and bidding systems were introduced. Every convention is based on certain assumptions, which are commonly referred to as "bridge laws". They provide an analogy for mathematical theorems, but even those that have long been regarded by the bridge community as true, generally are based more on experience, observations, or intuitions of bridge players, and not on scientific foundations.

The aim of this study is to check whether these "bridge laws" (treated as hypotheses) can be verified on the basis of statistical research. This would build a scientific basis which could then be used to construct new conventions and verification of existing ones. In addition to testing of already known hypotheses, some new “laws” will be tested, as well as connections between them. It also describes the statistical methods of solving the problems of decision making (that can happen during bridge auction), and a new method of verifying the randomness of a given set of deals. Project built for the purpose of this study consists of: deal generator (to generate deals with specific restrictions), a double dummy solver (by Piotr Beling), and tools for statistical calculations.

The thesis has been divided into four parts. The first consists of chapters 2. (where basic concepts used in the work are defined) and 3. (where current state of research and similar existing solutions were presented). The second part is section 4.1., which describes the solutions used in the project. In the third part (4.2.), experiments have been described: first six of them concerning hypotheses verification ("bridge laws"), other three are examples of solving decision problems in bridge auction. Results of the experiments were presented in the last part (chapter 5.).

Spis treści

1. Wstęp

Badania nad teorią brydża same w sobie nie są oficjalną dziedziną nauki. Jednak pojawienie się i rozwój brydża sportowego spowodowały, że z czasem został on uznany za jedną z dyscyplin sportów umysłowych. Status dyscypliny sportowej powoduje zaś, że obecnie można zajmować się naukowo-teoretycznymi aspektami brydża sportowego, podobnie jak na Akademii Wychowania Fizycznego naukowcy zajmują się tradycyjnymi dyscyplinami sportowymi np. szermierką, siatkówką, czy kajakarstwem.

Chociaż pierwsza Olimpiada Sportów Umysłowych odbyła się dopiero w 2008 roku (w Pekinie, po Letnich Igrzyskach Olimpijskich), ale za dyscyplinę sportu, którą można było się zajmować w sposób naukowy, nieoficjalnie uznano brydża już znacznie wcześniej. Konkretnie stało się to pod koniec lat sześćdziesiątych XX wieku, kiedy to w USA tworzono drużynę „Asów z Dallas", której głównym zadaniem miało być pokonanie włoskiego „Blue Teamu". Wtedy to, dla każdej z par wchodzących w skład drużyny, powstały nowe systemy licytacyjne, które skonstruowano w oparciu o badania naukowe. Zrobiono to w celu zapewnienia „Asom" jak największej przewagi na poziomie „technologicznym".

W Polsce już w latach sześćdziesiątych i siedemdziesiątych, badania naukowe nad teorią brydża prowadzili matematycy: Łukasz Sławiński oraz Stanisław Rumiński. Przy okazji tych badań sformułowanych zostało wiele zasad (postulatów) dotyczących konstrukcji systemów licytacyjnych, a ich efektem było powstanie zupełnie nowego (można by powiedzieć: „nieklasycznego") podejścia do licytacji, czyli tzw. „Systemów Słabych Otwarć" (inaczej: „silnopasowych"). Było to na tyle rewolucyjne podejście, że, jak twierdził Stanisław Rumiński: *Coraz większe sukcesy SSO powodowały natychmiastową kontrakcję. Najprostszym rozwiązaniem, jeżeli coś mnie pokonuje, jest zakaz. Przewaga tych systemów ujawniła się szczególnie w grze turniejowej, na co reakcją było niedopuszczenie „silnego pasa" do różnego rodzaju imprez brydżowych. Była to praktyka zupełnie odwrotna niż w innych sportach. Krótko mówiąc, była ona bardziej konserwatywna niż sam konserwatyzm*[1]. Zanim do tego doszło, i tak większą popularność zyskały jedynie w Polsce, i to tylko przez pewien okres czasu, chociaż nawet obecnie można niekiedy spotkać pary grające „SSO", o ile oczywiście takie systemy są dopuszczone do użytku (tak jest np. w rozgrywkach polskiej ligi).

1 Stanisław Rumiński, *Systemy Słabych Otwarć*, Warszawa 1984, s. 1.

Obecnie prace naukowe poświęcone teorii brydża nie są już wyjątkowym zjawiskiem. Od początku XXI wieku, brydżowi zostały poświęcone prace magisterskie: Piotra Belinga (*Praktyczne aspekty programowania gier logicznych*[2]), Pawła Perza (*Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*[3], 2009), czy Pawła Jassema (*Podejmowanie decyzji brydżowych za pomocą logiki rozmytej i metod regułowych*[4]).

1.1. Teza pierwsza – prawa brydżowe

Nie wiadomo, jakie były pierwsze prawa brydżowe, jednak do tej pory powstało już bardzo wiele mniej i bardziej poważnych mądrości. W czasach, w których powstawały, ich twórcy mogli polegać jedynie na własnych obserwacjach i intuicjach.

W niniejszej pracy chcę wykazać, że prawdziwość praw brydżowych można zweryfikować w oparciu o badania statystyczne, traktując je jako pewne hipotezy.

Prawa brydżowe na ogół są fundamentem, na którym budowane są konwencje oraz systemy licytacyjne. Najprostsze przykłady takich praw to: „lepiej jest, gdy rozgrywka prowadzona jest z ręki silniejszej", albo: „rozgrywka w kolor rozłożony 4-4 jest bardziej korzystna, niż w kolor rozłożony 5-3". Uważam, że badania statystyczne przeprowadzone na wygenerowanych rozdaniach spełniających odpowiednie warunki pozwolą potwierdzić, lub odrzucić tego rodzaju brydżowe hipotezy.

Poza badaniem pojedynczych praw, możliwe powinno być również badanie zależności między nimi, czyli ustalanie które prawo jest „mocniejsze". W matematyce, gdy jakieś twierdzenie zostanie dowiedzione, wtedy można formułować kolejne twierdzenia oparte na wcześniej już dowiedzionych. Analogicznie, na podstawie pozytywnie zweryfikowanych praw brydżowych można formułować kolejne prawa, i również próbować ich weryfikacji, w ten sposób zwiększając wiedzę na temat teoretycznych podstaw brydża sportowego. Przykładem takiego prawa może być: „rozgrywka w kolor rozłożony 4-4 jest bardziej korzystna, niż w kolor rozłożony 5-3, nawet gdy jest prowadzona ze słabszej ręki". Taka hipoteza implikuje, że prawo „silniejszej ręki" jest „słabsze", niż prawo „lepiej grać w kolor 4-4, niż 5-3".

Należy pamiętać, że faktyczny dowód (polegający na zbadaniu wszystkich rozdań o parametrach określonych w warunkach eksperymentu) jest teoretycznie możliwy, ale w

2 Piotr Beling, *Praktyczne aspekty programowania gier logicznych*, Łódź 2006.
3 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009.
4 Paweł Jassem, *Podejmowanie decyzji brydżowych za pomocą logiki rozmytej i metod regułowych*, Poznań 2014.

praktyce niewykonalny (z uwagi na ograniczenia wynikające z dostępnej mocy obliczeniowej i czasu potrzebnego na przeprowadzenie takiego dowodu). Z tego powodu w celu weryfikacji hipotez wykorzystane zostaną metody statystyczne. Hipoteza zweryfikowana pozytywnie przy użyciu badań statystycznych w dalszym ciągu może teoretycznie być fałszywa (prawdopodobieństwo zajścia takiej sytuacji jest tym mniejsze, im większa była próba).

1.2. Teza druga – problemy decyzyjne

W trakcie licytacji brydżowej, gracze często muszą rozwiązywać problemy decyzyjne. Zazwyczaj pojawiają się one wtedy, gdy kończą się uzgodnione sekwencje opisane w systemie licytacyjnym, na przykład w trakcie licytacji obustronnej, w której większość par ma jedynie ustalenia o bardzo ogólnym charakterze.

Przykładowe problemy decyzyjne, jakie można napotkać w trakcie licytacji, są następujące: 1) pasować, czy kontrować, 2) kontrować, czy licytować, 3) pasować, czy licytować, oraz 4) pasować, kontrować, czy licytować. Przy rozwiązywaniu tych problemów często nie można odwołać się do żadnych praw brydżowych, a jedynie do własnych doświadczeń i intuicji. Wybrane problemy tego typu (z założenia trudne) pojawiają się często w konkursie, publikowanym na łamach miesięcznika „Brydż”. Później publikowane są także odpowiedzi, opinie i komentarze ekspertów oraz czytelników. Nawet eksperci często udzielają bardzo różnych odpowiedzi, co oznacza, że w przypadku trudnych problemów intuicja może zawodzić także doświadczonych brydżystów.

W niniejszej pracy chcę wykazać, że tego rodzaju problemy można rozwiązywać przy użyciu metod statystycznych, tzn. wyniki tych badań pozwalają ustalić, która decyzja jest najbardziej korzystna z punktu widzenia zapisu brydżowego.

Pewnym problemem będzie interpretacja dotychczasowego przebiegu licytacji, która musi być jednoznaczna i poprawna. Błędna interpretacja spowoduje, że eksperyment zostanie wykonany w oparciu o fałszywe założenia i zwrócone wyniki będą bezwartościowe.

1.3. Struktura pracy

Praca została podzielona na sześć rozdziałów (wstęp jest pierwszym z nich). W rozdziale drugim omówione zostaną podstawy teoretyczne: terminologia i zapis brydżowy,

pojęcia wprowadzone na potrzeby pracy, algorytmy, oraz wzory i pojęcia z dziedziny statystyki. W trzecim rozdziale nakreślony zostanie dotychczasowy stan prac. Czwarty rozdział składa się z dwóch podrozdziałów: w pierwszym z nich opisany zostanie projekt magisterski, natomiast w drugim omówione zostaną przeprowadzone eksperymenty. W rozdziale piątym zaprezentowane zostaną rezultaty badań, natomiast ostateczne wnioski na temat prawdziwości postawionych we wstępie tez, opisane będą w rozdziale szóstym (zakończeniu), podobnie jak wskazanie dalszych potencjalnych kierunków badań.

2. Podstawy teoretyczne

2.1. Terminologia brydżowa

2.1.1. Podstawowe pojęcia

W brydża gra się za pomocą standardowej talii, składającej się z 52 unikalnych kart. Każda karta ma jeden z czterech kolorów brydżowych, oraz jedno z trzynastu mian.

Kolor karty określa pojawiający się na niej symbol: ♠ (pik), ♥ (kier), ♦ (karo) lub ♣ (trefl). Kolory dzielą się na starsze (piki oraz kiery) i młodsze (kara oraz trefle). Tradycyjnie przyjęło się, że piki i trefle są czarne, natomiast kiery i kara – czerwone, natomiast we współcześnie produkowanych taliach zdarza się, że faktyczna barwa przyporządkowana każdemu z kolorów brydżowych może być inna (np. trefle są zielone, a kara – różowe).

Miano karty określa liczba (od 2 do 10) lub litera (A (as), K (król), D (dama), W (walet) – w przypadku kart polskich, A, K, Q, J – w przypadku kart angielskich).

Figura to karta, której miano jest określane przez literę, np. ♥K (król).

Honor to karta, której miano nie jest jednocyfrową liczbą, np. ♥10 jest honorem, ♥3 nie jest honorem.

Blotka, to karta, której miano jest jednocyfrową liczbą, np. ♥2 jest blotką, ♥K nie jest blotką.

Punkty za honory (figury)[5], to przyjęte wartości przyporządkowane figurom. Obecnie najbardziej rozpowszechniony sposób liczenia punktów jest następujący: as jest wart cztery punkty, król – trzy punkty, dama – dwa punkty, walet – jeden punkt. Siła ręki gracza jest równa sumie punktów za posiadane przez niego figury.

Przykład: *Gracz ma w swoim posiadaniu następujące figury: ♠K, ♥A, ♦KD oraz ♣AW. Siła ręki gracza jest równa: 3 + 4 + 3 + 2 + 4 + 1 = 17 punktów honorowych.*

Rozdanie brydżowe to pojedyncza rozgrywka brydżowa, rozpoczynająca się od rozdzielenia kart dla czterech graczy[6], licytacji oraz rozgrywki[7]. Rozdanie kończy się ustaleniem zapisu, będącego efektem licytacji i rozgrywki.

---

5 Tradycyjnie przyjęło się pojęcie „punkty honorowe”, ale „10” chociaż jest honorem, warta jest zero punktów, dlatego bardziej precyzyjne byłoby stworzone przez analogię pojęcie „punkty figurowe”, które jednak w niniejszej pracy nie będzie stosowane.

6 W trakcie meczu lub turnieju, karty na ogół nie są rozdawane przez graczy, lecz trafiają na stół w specjalnych pudełkach z przegródkami, z których gracze wyciągają karty odpowiadające zajmowanym przez nich pozycjom.

7 Za wyjątkiem sytuacji, gdy w licytacji wszyscy gracze spasowali.

Pozycja gracza to jedna z czterech przyjętych nazw: West (S), North (N), East (E), South (S). Na ogół używa się skrótów: W, N, E, S. „S zagrał ♥K” oznacza, że gracz zajmujący pozycję South zagrał króla kier. W grze towarzyskiej pozycje graczy są ustalane dowolnie, przy czym gracze N i S muszą siedzieć naprzeciwko siebie (podobnie jak gracze W i E), natomiast w turniejach i meczach pozycje graczy zostają odczytane z pudełka, w którym przekazywane są (wcześniej rozdane) karty. Gracze N i S stanowią jedną parę, natomiast gracze W i E – drugą. Używane są również pojęcia „linia NS” oraz „linia WE”, które odnoszą się także do rozkładu kart na tych dwóch pozycjach.

Ręka to trzynaście kart, które zostały rozdane jednemu z graczy.

Przykład: *♠AK105 ♥D32 ♦KD108 ♣52 to ręka gracza S.*

Rozkład brydżowy to cztery uporządkowane ręce, z których każda została przypisana do innej pozycji.

Ograniczenia układu ręki to ustalone minimalne i maksymalne liczby kart w każdym z kolorów. Minimalnym możliwym ograniczeniem jest 0, a maksymalnym 13.

Przykład: *W ręce S może się znajdować od 3 do 5 kierów.*

Ograniczenia siły ręki to ustalona minimalna i maksymalna liczba punktów za honory, które mogą się znajdować w danej ręce. Minimalnym możliwym ograniczeniem jest 0, a maksymalnym 37.

Przykład: *Gracz S może mieć od 15 do 17 punktów.*

Kolor rozłożony X – Y, to taki kolor, w którym jeden z graczy ma X kart w tym kolorze, natomiast jego partner ma Y kart w tym kolorze.

Przykład: *Na linii NS, kiery są rozłożone 5-4*.

2.1.2. Pojęcia związane z licytacją i rozgrywką

Licytacja to część rozdania, w której ustalana jest ręka rozgrywającego, kolor atutowy (lub jego brak) oraz liczba lew, którą rozgrywający musi wziąć, aby jego linia uzyskała zapis.

Odzywka licytacyjna to zapowiedź dana przez gracza w trakcie licytacji. Istnieją 3 odzywki o znaczeniu specjalnym: pas, kontra i rekontra oraz 35 odzywek zwykłych, z których każda składa się z liczby od 1 do 7 oraz koloru (lub bez atu).

Rozgrywający to gracz zajmujący pozycję, z której po raz pierwszy został zalicytowany kolor w jaki jest rozgrywany kontrakt, należący do pary, która wygrała licytację. Po rozpoczęciu rozgrywki przez wistującego, karty należące do partnera

rozgrywającego zostają wyłożone odkryte na stole i do końca rozgrywki dysponuje nimi rozgrywający.

Wistujący to gracz, który zajmuje pozycję po lewej stronie rozgrywającego. Jego zagranie (**wist**, zwany też czasem **pierwszym wyjściem**) rozpoczyna rozgrywkę.

Lewa to cztery karty, z których każda została dołożona przez innego z graczy. Do karty wyjścia, według ruchu wskazówek zegara dokładane są karty z kolejnych rąk. Wyjście następuje z ręki, która wzięła poprzednią lewę lub, w przypadku gdy jeszcze rozgrywka się nie rozpoczęła, z ręki wistującego. Istnieje obowiązek dokładania kart o kolorze zgodnym z kolorem karty wyjścia, a w przypadku gdy któraś z rąk nie dysponuje taką kartą, zamiast niej gracz może dołożyć dowolną inną kartę.

Kolor atutowy to kolor ostatniej odzywki zwykłej, która została dana w licytacji. W przypadku gdy ta odzywka została dana w bez atu, w rozgrywce nie ma koloru atutowego. Karty należące do koloru atutowego mają tę własność, że jeśli gracz nie ma karty w kolorze karty wyjścia, może dołożyć kartę z koloru atutowego. Taka lewa zostanie wzięta przez rękę, która dołożyła do niej najwyższą kartę koloru atutowego.

Przykład 1. *Ostatnią odzywką zwykłą w licytacji było 4♠, zatem kolorem atutowym są piki.*

Przykład 2. *Ostatnią odzywką zwykłą w licytacji było 3BA, zatem nie będzie koloru atutowego.*

Przykład 3. *Gracz W wyszedł ♣A, przeciwko kontraktowi 4♥. Z rąk N i E zostały dołożone ♣2 i ♣3, gracz S nie miał żadnego trefla i dołożył ♥2, dzięki czemu wziął lewę, ponieważ kiery są kolorem atutowym.*

Rozgrywka z ręki X oznacza, że rozgrywający zajmuje pozycję X.

Przykład: *Kontrakt 4♠ jest rozgrywany z ręki S, zatem S jest rozgrywającym, natomiast W wistującym.*

Otwarcie to pierwsza odzywka licytacyjna nie będąca pasem.

Przykład: *W licytacji „pas pas 1♥ pas 1BA", odzywka 1♥ jest otwarciem*.

Zgłoszenie koloru to odzywka licytacyjna, za pomocą której gracz po raz pierwszy przekazuje informację o tym, że posiada określoną liczbę kart w konkretnym kolorze.

Przykład: *S otwiera 1♠, informując o posiadaniu co najmniej pięciu kart w kolorze pikowym, czyli zgłasza ten kolor*.

Poparcie koloru to odzywka licytacyjna, która przekazuje informację o tym, że gracz dysponuje określoną liczbą kart (na ogół trzema lub więcej), w kolorze zgłoszonym przez partnera.

Przykład: *S otwiera 1♠ (które oznacza posiadanie co najmniej 5 pików), W pasuje, N licytuje 2♠, które oznacza co najmniej trzy karty w pikach, więc 2♠ jest poparciem koloru.*

Równorzędne kolory, to takie, że w każdym z nich partnerzy mają łącznie tyle samo kart.

Przykład: *S ma 4 piki i 3 kiery, N ma 4 piki i 5 kierów, razem mają 8 pików i 8 kierów, zatem dla pary NS piki i kiery są kolorami równorzędnymi.*

Kontrakt definiowany jest przez ostatnią odzywkę zwykłą w licytacji. Na kontrakt składają się: wysokość (liczba od 1 do 7) oraz kolor (piki, kiery, kara, trefle lub bez atu).

Przykład: *Gdy ostatnią odzywką zwykłą w licytacji było 3♠, to wysokość kontraktu wynosi 3, a kolorem atutowym są piki*.

Kontrakt zrealizowany jest wtedy, gdy rozgrywający wziął co najmniej tyle lew, ile wynosiła wysokość kontraktu zwiększona o sześć.

Przykład: *Aby zrealizować kontrakt 3BA, rozgrywający musi wziąć co najmniej 3 + 6 = 9 lew.*

Licytacja jednostronna to taka, w której tylko gracze należący do jednej pary zgłaszają odzywki licytacyjne inne niż „pas”.

Przykład: *Fragment licytacji przedstawiony w tabeli 2.1 to licytacja jednostronna*.

Tabela 2.1

W	N	E	S
pas	1♣	pas	1♥
pas	2♦	pas	3♣
pas	3♥	pas	...

Pas to odzywka specjalna, która na ogół oznacza brak zainteresowania udziałem w licytacji. Trzy pasy kończą licytację, chyba że jeden z graczy nie miał jeszcze okazji dać żadnej odzywki licytacyjnej – w takim przypadku licytację kończą cztery pasy.

Kontra to odzywka specjalna, która w przypadku gdy ostatnia odzywka zwykła stanie się kontraktem, podwaja zapis uzyskany za realizację tego kontraktu lub karę w przypadku gdy nie zostanie on zrealizowany. Kontrę może dać tylko gracz należący do pary, która nie zapowiedziała kontrowanej odzywki.

Rekontra to odzywka specjalna, która w przypadku, gdy ostatnia odzywka zwykła stanie się kontraktem i została skontrowana, czterokrotnie zwiększa zapis uzyskany za realizację tego kontraktu lub karę w przypadku, gdy nie zostanie on zrealizowany.

Rekontrę może dać tylko gracz należący do pary, która zapowiedziała ostatnią odzywkę zwykłą, i tylko wtedy gdy została ona skontrowana.

Słabsza ręka to ta ręka należąca do danej pary, która zawiera mniej punktów honorowych.

Silniejsza ręka to ta ręka należąca do danej pary, która zawiera więcej punktów honorowych.

Rozgrywka w otwarte karty to taka rozgrywka, w której karty żadnego z graczy nie są zakryte, czyli wszyscy gracze dysponują pełną informacją na temat rozkładu kart. Na ogół stosuje się ją w problemach teoretycznych w celu zobrazowania manewrów rozgrywającego bądź obrońców, oraz w programach brydżowych (algorytm Min-Max, opisany w podrozdziale 2.3).

Rozkład lew to wektor 14 – elementowy, w którym kolejne wartości oznaczają prawdopodobieństwo wzięcia określonej liczby lew, przy rozgrywce z ręki ustalonego gracz w ustalony kolor. Te prawdopodobieństwa zostają wyznaczone na podstawie wyników rozgrywki w otwarte karty wielu rozdań. Będzie wykorzystywany do rozwiązywania problemów brydżowych, które wymagają obliczenia wartości oczekiwanych dla poszczególnych decyzji.

Przykład: *Rozegrano 10 rozdań z ręki gracza S, w piki. W dwóch rozdaniach wzięto 5 lew, w trzech: 6 lew, w jednym 7 lew, w dwóch 8 lew i w jednym 9 lew. Rozkład lew wygląda w następujący sposób (W – kolejne elementy wektora, L – odpowiadające im liczby lew):*

W	0	0	0	0	0	0,2	0,3	0,2	0,2	0,1	0	0	0	0
L	0	1	2	3	4	5	6	7	8	9	10	11	12	13

Problem brydżowy (decyzyjny) to sytuacja, w której dysponujemy pewnym zbiorem informacji na temat rozdania i powstaje konieczność podjęcia jakiejś decyzji (na ogół najbardziej korzystnej z perspektywy zapisu brydżowego), w oparciu o te informacje.

Przykład: *Z licytacji wynika, że N dysponuje siłą 15-17 punktów i ma co najmniej cztery kiery. S posiada 10 punktów oraz cztery kiery i ma do podjęcia decyzję: czy zalicytować 4♥, czy może 3BA.*

2.1.4. Zapis brydżowy

Zapis brydżowy to metoda obliczania punktacji za pojedyncze rozdania, zależna od kontekstu. Kontekst ten wynika z rezultatów wcześniej rozegranych rozdań (w brydżu robrowym), lub jest narzucony w sposób sztuczny (w brydżu sportowym).

Założenia to kontekst w jakim grane jest rozdanie. Istnieją cztery rodzaje założeń: obie przed partią, NS po partii, WE po partii, obie po partii.

Punktacja podstawowa to liczba zależna od koloru i wysokości kontraktu, oraz kontry i rekontry. Za każdy poziom kontraktu zrealizowanego w kolor młodszy liczy się 20 punktów, a w kolor starszy lub bez atu – 30 punktów, za wyjątkiem pierwszego poziomu w bez atu, za który wyjątkowo liczy się 40 punktów. Kontra podwaja ten zapis, rekontra podwaja jeszcze raz.

Przykład 1: *Grany jest kontrakt 2BA. Punktacja podstawowa za ten kontrakt wynosi 40 (za pierwszy poziom) + 30 (za drugi) punktów*.

Przykład 2: *Grany jest kontrakt 6♠. Punktacja podstawowa za ten kontrakt wynosi 6 x 30 = 180 punktów*.

Przykład 3: *Grany jest kontrakt 3♦ z kontrą. Punktacja podstawowa za ten kontrakt wynosi (3 x 20) x 2 = 120 punktów*.

Przykład 4: *Grany jest kontrakt 1♣ z rekontrą. Punktacja podstawowa za ten kontrakt wynosi (1 x 20) x 4 = 80 punktów*.

Założenia **„obie przed partią”**, oznaczają, że obie pary są przed partią.

Założenia **„NS po partii”** oznaczają, że para NS jest po partii, a WE przed partią.

Założenia **„WE po partii”** oznaczają, że para WE jest po partii, a NS przed.

Założenia **„obie po partii”** oznaczają, że obie pary są po partii.

Nadróbki, to lewy wzięte powyżej liczby lew wynikającej z wysokości zadeklarowanego kontraktu.

Przykład: *S grał 3♥, ale wziął 11 lew. Oznacza to, że S wziął 2 nadróbki.*

W opisywanym tutaj eksperymencie wyniki obliczane są według **punktacji sportowej**. Oznacza to, że do każdego kontraktu, za który punktacja podstawowa nie przekracza 100 punktów, doliczana jest specjalna premia o wysokości 50 punktów. W przypadku, gdy punktacja podstawowa przekracza 100 punktów, premia wynosi 300 punktów (jeśli strona rozgrywająca była przed partią) lub 500 punktów (jeśli strona rozgrywająca była po partii). Za kontrakty zrealizowane na wysokości 6 oraz 7 są dodatkowe premie: na poziomie 6 jest to 500 punktów przed partią (750 po partii), a na poziomie 7: 1000 (1500). Ponadto doliczane są również dodatkowe premie za nieudaną kontrę (50), lub udaną rekontrę (100). Za nadróbki dodaje się punkty zależne od koloru (tak jak przy obliczaniu punktacji podstawowej), chyba że była kontra – w takim przypadku, niezależnie od koloru każda nadróbka dodaje 100 punktów (przed partią) lub 200 (po partii), w przypadku rekontry te wartości zostają podwojone.

Przykład 1: *S grał 4♥ z kontrą (przed partią) i wziął 11 lew. Zapis dla strony NS to (4 x 30) x 2 (zapis podstawowy) + 50 (nieudana kontra) + 300 (zapis podstawowy 100 lub większy) + 100 (za jedną nadróbkę) = 690*.

Przykład 2: *W grał 2♦ z rekontrą (przed partią) i wziął 9 lew. Zapis dla strony WE to (2 x 20) x 4 (zapis podstawowy) + 300 (zapis podstawowy 100 lub większy) + 100 (udana rekontra) + 2 x 100 (nadróbka z rekontrą) = 760*.

Przykład 3: *N grał 6♠ z kontrą (po partii) i wziął 13 lew. Zapis dla NS to (6x 30) x2 (zapis podstawowy) + 50 (nieudana kontra) + 500 (zapis podstawowy 100 lub większy) + 750 (premia za kontrakt na poziomie 6) + 200 (nadróbka z kontrą po partii) = 1860.*

Jeżeli kontrakt nie zostanie zrealizowany, zapis uzyskują przeciwnicy i jest on zależny od liczby lew, których zabrakło (wysokości tzw. **wpadki**). Przed partią, każda brakująca lewa oznacza 50 punktów dla przeciwników, po partii – 100. W przypadku, gdy kontrakt został skontrowany, przed partią kolejne lewy wpadkowe są warte: pierwsza – 100, druga i trzecia po 200, czwarta i kolejne po 300, natomiast po partii jest to: 200 za pierwszą i trzysta za każdą następną. Rekontra podwaja ten zapis.

Przykład 1: *S grał 3BA (po partii) i wziął 6 lew. Strona WE zapisuje 3 x 100 = 300 punktów.*

Przykład 2: *S grał 2♠ z kontrą (przed partią) i wziął 6 lew. WE zapisują 1x 100 + 1 x 200 = 300 punktów.*

Przykład 3: *W grał 4♣ z rekontrą (przed partią) i wziął 6 lew. NS zapisują: (1x 100 + 2x 200 + 1x 300) x 2 = 1600.*

2.1.5. Pojęcia wprowadzone na potrzeby projektu

Wektor wyników to 14- elementowy wektor wyznaczany dla każdego kontraktu, w którym każdy element jest wynikiem uzyskiwanym w przypadku wzięcia kolejno od 0 do 13 lew.

Przykład: *Dla kontraktu 3BA, przed partią, bez kontry, wektor wyników to [-450, -400, -350, -300, -250, -200, -150, -100, -50, 400, 430, 460, 490, 520].*

Reprezentacja talii w projekcie to przyporządkowanie każdej karcie liczby całkowitej z przedziału od 1 do 52. Jest to funkcja różnowartościowa, która została przedstawiona w tabeli 2.2.

Tabela 2.2

	A	**K**	**D**	**W**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**
♠	1	2	3	4	5	6	7	8	9	10	11	12	13
♥	14	15	16	17	18	19	20	21	22	23	24	25	26
♦	27	28	29	30	31	32	33	34	35	36	37	38	39
♣	40	41	42	43	44	45	46	47	48	49	50	51	52

Reprezentacja ręki to uporządkowany wektor 13 liczb, które odpowiadają kartom według tabeli 2.2.

Przykład: *[1, 3, 5, 6, 17, 18, 20, 21, 27, 28, 30, 33, 47] to reprezentacja ręki: ♠AD109 ♥W1087 ♦AKW8 ♣7.*

Reprezentacja rozdania to uporządkowany wektor reprezentacji czterech rąk: [ręka S, ręka W, ręka N, ręka E].

Zestaw ograniczeń to uporządkowany wektor par, z których każda opisuje zakres wartości: [punkty, piki, kiery, kara, trefle], a każda z tych informacji zapisana jest w formie [min, max].

Przykład: *[[12, 17], [4, 5], [3, 4], [2, 3], [1, 4]] to zestaw ograniczeń dla ręki N. Oznacza to, że w ręce N jest od 12 do 17 punktów, 4 lub 5 pików, 3 lub 4 kiery, 2 lub 3 kara, oraz od 1 do 4 trefli.*

Wariant ograniczeń to jeden z zestawów ograniczeń, do którego może zostać dopasowana ręka. Z tego powodu parametry przekazywane jako ograniczenia dla każdej z rąk podawane są w formie listy wszystkich akceptowanych wariantów. Aby wygenerowane rozdanie zostało zwrócone jako poprawne, do każdej ręki musi zostać dopasowany przynajmniej jeden z wariantów.

Przykład: *Zestaw ograniczeń dla ręki pasującej do otwarcia 1♦ w systemie licytacyjnym „Wspólny Język" składa się z sześciu wariantów: 1) 12-14 punktów, co najmniej 5 kar, 2) 12-17 punktów, układ 5 trefli i cztery kara, 3) 15-17 punktów, układ niezrównoważony[8] i co najmniej 5 kar, 4) 12-17 punktów układ 4 kara, 4 kiery, 4 piki, 1trefl, 5) 12-17 układ 4 kara, 4 kiery, 4 trefle i 1 pik lub 6) 12-17 punktów, układ 4 kara, 4 piki, 4 trefle, 1 kier.*

Zadane parametry to zbiór wszystkich informacji przekazywanych do funkcji generatora rozdań z ograniczeniami. Mogą to być zestawy ograniczeń dla

8 Układ niezrównoważony to taki, który w jednym z kolorów ma 0-1 kartę, lub ma więcej niż jeden kolor 2 – kartowy. Przykład: 5431, 5530, 4441, 5422 to układy niezrównoważone.

poszczególnych rąk albo znane ręce (tzn. informacje o tym, że gracz posiada konkretną rękę).

Przykład: *Ręka S to ♠AKW10 ♥KDW10 ♦AW5 ♣54, W ma 10-12 punktów i sześć kierów, N ma 7-10 punktów i cztery piki, E ma 0-6 punktów i sześć trefli. Te informacje reprezentowane w odpowiedni sposób to zadane parametry.*

Reprezentacja parametrów to forma w jakiej zadane parametry są przekazywane do generatora rozdań (pojęcie generatora rozdań zostanie zdefiniowane w podrozdziale 2.2). W przypadku, gdy któraś z rąk jest znana, zostaje ona przekazana jako reprezentacja ręki. Jeśli są to ograniczenia, należy je przekazać w formie wektora wariantów: [$wariant_1$, $wariant_2$, …, $wariant_n$], gdzie każdy wariant jest reprezentowany przez odpowiedni zestaw ograniczeń.

Konwencja brydżowa (licytacyjna) to ustalenie graczy należących do tej samej pary, które dotyczy znaczenia określonej sekwencji odzywek w licytacji. Ustalenia takie mają charakter jawny (czyli przeciwnicy również mają prawo znać ich znaczenie). Na ogół konwencje oparte są prawach brydżowych.

Przykład: *Po otwarciu 1BA, jeżeli para gra konwencją znaną jako „transfer", odpowiedzi 2♦ oraz 2♥ nie oznaczają posiadania odpowiednio kar i kierów, lecz co najmniej pięciu kierów (2♦) lub pików (2♥). Konwencja ta jest oparta na założeniu, że dla pary bardziej korzystne jest, gdy rozgrywka prowadzona jest z silniejszej ręki, dlatego gracz otwierający 1BA po „transferze" poprawi na właściwy kolor.*

Prawo brydżowe to potoczne pojęcie, które oznacza na ogół jakąś mniej lub bardziej popularną w świecie brydżystów regułę, do której mogą istnieć odwołania w literaturze. Przyjmujemy założenie, że każde z takich praw da się przedstawić w formie możliwej do zweryfikowania hipotezy.

Przykład: *Jeżeli para posiada dwa równorzędne kolory, z których jeden jest rozłożony 5-3 , a drugi 4-4, to ta para ma do wzięcia więcej lew, gdy kolor 4-4 będzie kolorem atutowym.*

Rozkład lew w rozdaniu to informacja na temat tego, ile lew może wziąć każdy z graczy przy rozgrywce w z jego ręki w każdy z kolorów oraz w bez atu. Informacja ta zostaje uzyskana za pomocą algorytmu Min-Max (więcej na temat tego algorytmu w podrozdziale 2.3).

Rozkład lew w eksperymencie to średni rozkład lew we wszystkich rozdaniach rozegranych w ramach tego eksperymentu. Eksperymenty związane z badaniem praw brydżowych na ogół polegają na uzyskaniu tego rozkładu dla możliwie dużej liczby rozdań

spełniających zadane parametry. Rozkład ten będzie służył bezpośrednio do weryfikacji prawa brydżowego.

2.2. Generowanie rozdań – algorytmy

Algorytm deterministyczny, to algorytm, którego działanie jest całkowicie zdeterminowane przez warunki początkowe (wejście). Oznacza to, że kilkukrotne uruchomienie takiego algorytmu z takim samym wejściem, doprowadzi za każdym razem do takiego samego wyniku.

Przykład: *Algorytm zwracający moduł liczby X (jeśli X > 0, zwróć X, w przeciwnym przypadku zwróć -X) to algorytm deterministyczny.*

Algorytm niedeterministyczny*,* to algorytm, który dla tego samego wejścia może przeprowadzić obliczenia na wiele sposobów i zwracać różne wyjścia.

Algorytm probabilistyczny, to algorytm, który do swojego działania stosuje losowość (czyli w praktyce przy obliczeniach wykorzystuje generator liczb losowych). Algorytmy tego typu z pewnym prawdopodobieństwem mogą zwracać niepoprawne wyjścia.

Przykład: *Test pierwszości Fermata wykonuje określoną liczbę prób, z których każda polega na tym, że losowana jest liczba z przedziału od 1 do zadanej liczby, a następnie szukany jest największy wspólny dzielnik dla tych liczb (podanej na wejściu i wylosowanej). Jeśli w dowolnej próbie NWD będzie różne niż 1, test zwróci informację, że liczba podana na wejściu nie jest liczbą pierwszą, natomiast po osiągnięciu podanej liczby prób zwróci informację, że liczba podana na wejściu jest liczbą pierwszą, co nie musi wcale być informacją prawdziwą (prawdopodobieństwo tego, że jest prawdziwa, jest tym większe, im więcej przeprowadzono prób). Zatem test Fermata jest algorytmem probabilistycznym.*

Algorytm Las Vegas to algorytm korzystający z losowości, który zawsze zwraca poprawne wyjście lub informację na temat niepowodzenia (losowość nie ma wpływu na poprawność wyniku, dotyczy jedynie sposobu przeprowadzania obliczeń). Algorytmy tego typu mogą być używane w sytuacjach, gdzie liczba możliwych rozwiązań jest względnie ograniczona, a weryfikacja poprawności danego rozwiązania jest względnie łatwa, natomiast obliczenie rozwiązania jest skomplikowane.

Przykład: *Algorytm szybkiego sortowania quicksort polega na tym, że z tablicy wybierany jest element rozdzielający (w sposób losowy), po czym tablica zostaje podzielona na dwa fragmenty: do początkowego przenoszone są wszystkie elementy nie*

większe od rozdzielającego, a do końcowego wszystkie większe. Następnie taka sama procedura uruchamiana jest dla początkowej i końcowej części tablicy. Rekurencja kończy się wtedy, gdy kolejny uzyskany z podziału fragment zawiera pojedynczy element. Zawsze zwracana jest posortowana tablica.

Metoda Monte Carlo to algorytm korzystający z losowości, którego czas działania jest ograniczony z góry, ale może on zwracać niepoprawne wyjście z pewnym (na ogół niskim) prawdopodobieństwem. Algorytm tego typu można przekształcić w algorytm Las Vegas, w taki sposób, żeby powtarzać działanie metody Monte Carlo aż do momentu, gdy zwrócone zostanie wyjście, które spełnia ustalone warunki.

Przykład: *Algorytm obliczania liczby* π *za pomocą metody Monte Carlo. Mając kwadrat o długości boku* $2r$ *oraz okrąg o promieniu* r *wpisany w ten kwadrat, wykonujemy* n *prób, z których każda polega na wylosowaniu dwóch liczb z zakresu od 0 do* $2r$ *, a następnie sprawdzeniu, czy punkt o wylosowanych w ten sposób współrzędnych znajduje się wewnątrz okręgu. Liczbę* π *wyznaczamy poprzez podzielenie liczby trafień przez liczbę prób, a następnie pomnożenie przez 4 (ponieważ pole kwadratu równe jest* $4r^2$ *).*

Generator rozdań to algorytm, który zwraca podział talii kart na cztery trzynasto-elementowe podzbiory.

Generator rozdań losowych to generator rozdań, w którym zwracany podział talii nie jest możliwy do przewidzenia, a generowane rozdania są zgodne z rozkładami matematycznymi (wynikającymi z rachunku prawdopodobieństwa).

Generator rozdań z ograniczeniami to taki generator rozdań, w którym ustalone ręce spełniają konkretne warunki podawane na wejściu (warunki mogą być określone w różny sposób – może to być podany z góry podzbiór kart należących do wybranej ręki lub rąk, informacja o tym w jakim przedziale powinna mieścić się liczba punktów za honory albo liczba kart w wybranym kolorze w ręce gracza).

Generator rozdań z ograniczeniami wykorzystujący metodę Monte Carlo, to generator rozdań z ograniczeniami, który na wejściu otrzymuje liczbę prób, po których ma zakończyć działanie, jeśli żadna z nich się nie powiedzie. Każda z prób polega na wygenerowaniu losowego rozdania i sprawdzeniu, czy spełnia ono zadane ograniczenia – jeśli tak, algorytm zwraca takie rozdanie i kończy działanie, w przeciwnym przypadku podejmuje kolejną próbę. Jeśli żadna próba nie zakończyła się sukcesem, algorytm zwraca informację o niepowodzeniu (algorytmy opisane w podrozdziale 4.1 z przyczyn technicznych zwracają zawsze dwie wartości: pierwszą z nich jest wygenerowane

rozdanie, a drugą jest informacja o powodzeniu). Taki algorytm jest obarczony jednostronnym błędem (to, że nie udało się wygenerować rozdania spełniającego zadane warunki, wcale nie musi oznaczać, że takie rozdanie nie istnieje).

Generator rozdań z ograniczeniami oparty na algorytmie Las Vegas, to generator rozdań z ograniczeniami, który dla każdego poprawnego wejścia zwraca poprawne wyjście, natomiast nie można ustalić czasu jego działania. Algorytm ten generuje kolejne losowe rozdania aż do momentu, gdy wylosowane rozdanie będzie spełniało podane na wejściu warunki. Istnieją metody pozwalające zoptymalizować jego działanie. Paweł Perz, autor pracy poświęconej generowaniu rozdań brydżowych, napisał: *Ilość prób, które są konieczne do wykonania rośnie bardzo szybko wraz ze zmniejszaniem się liczby pasujących układów. Powoduje to, że algorytm ten będzie mógł być efektywnie wykorzystywany tylko do generowania rozdań z dużą liczbą rozkładów sprzyjających*[9].

Algorytm grafowy to taki algorytm, który działa w oparciu o strukturę danych reprezentowaną w postaci grafu.

Generator rozdań z ograniczeniami oparty na algorytmie grafowym, to generator rozdań z ograniczeniami, który: *Opiera się na idei przedstawienia wszystkich rozdań brydżowych w postaci grafu G. Wierzchołkami grafu G są konkretne układy kart. Każdy wierzchołek sąsiaduje ze wszystkimi wierzchołkami (rozdaniami), które można uzyskać poprzez zamianę dwóch kart, pomiędzy różnymi rękami. (…) W tym algorytmie przeszukiwany będzie graf rozdań w poszukiwaniu rozdań pasujących do licytacji. Po ustaleniu ograniczeń, jakie rozdanie musi spełniać, aby pasować do licytacji, należy podać algorytm, który będzie pozwalał porównać dwa dowolne rozdania i wybrać z nich to, które jest bliżej zbioru szukanych rozwiązań (gdzie przez odległość rozumie się długość ścieżki w grafie G, pomiędzy rozdaniami)*[10].

W projekcie został wykorzystany algorytm generowania rozdań oparty na metodzie Monte Carlo, z uwagi na małą złożoność algorytmu (czas działania zależny jest jedynie od liczby prób wprowadzonej przez użytkownika), prostą implementację oraz wysoką skuteczność dla mało skomplikowanych ograniczeń (co czyni go bardzo wygodnym do testowania hipotez o charakterze ogólnym, czyli dotyczących praw brydżowych)[11]. Należy również podkreślić, że w przypadku gdy podana jest ręka jednego lub dwóch graczy, znacząco zmniejsza się liczba możliwych podziałów pozostałych kart, zatem nawet przy

9 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009, s. 43-44.
10 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009, s. 44-45.
11 Więcej informacji na temat tego, dlaczego nie został wybrany algorytm grafowy znajduje się w podrozdziale 3.1.

bardziej precyzyjnych ograniczeniach, algorytm wykazuje się wysoką skutecznością. Niewątpliwie słabością takiego algorytmu jest to, że w przypadku gdy żadna z rąk nie jest znana, bardzo precyzyjne ograniczenia mogą powodować, że prawie w ogóle nie będzie zwracał poprawnych rozdań (jeśli liczba ustalona liczba prób będzie zbyt niska), przez co konieczne staje się wprowadzenie specjalnego mechanizmu do wykrywania rozdań błędnych.

Ponieważ projekt zakłada generowanie i rozgrywanie rozdań, główny koszt jeśli chodzi o złożoność i tak przypada na algorytm Min-Max (ponad 99% czasu obliczeń), a sam czas generowania rozdania staje się pomijalny, i to niezależnie od zastosowanego w generatorze algorytmu. Problem generowania rozdań z ograniczeniami został już szeroko omówiony w pracy Pawła Perza[12], natomiast celem tej pracy nie było skupianie się na ich implementacji, lecz na prowadzeniu eksperymentów i weryfikacji hipotez w oparciu o rezultaty rozegranych rozdań.

2.3. Algorytm Min-Max

Piotr Beling, twórca programu *BridgeCalculator* wykorzystanego w projekcie, w artykule pt. *Efektywne rozwiązanie problemu rozgrywki w otwarte karty w brydżu*[13], napisał: *Rozwiązanie problemu gry w otwarte karty (ang. double dummy problem) polega na wyznaczeniu optymalnej linii rozgrywki przy założeniu pełnej wiedzy o rozkładzie (wszystkich) kart i bezbłędnej gry wszystkich uczestników. Ściślej, dla zadanej sytuacji, należy wyznaczyć liczbę lew, jaką weźmie każda ze stron*[14].

Algorytm Min-Max to algorytm oparty na metodzie minimalizowania maksymalnych możliwych strat (lub maksymalizacji minimalnego zysku), która wywodzi się z teorii gry o sumie zerowej.

Zastosowanie algorytmu Min-Max w rozgrywce brydżowej polega na tym, że gracze należący do dwóch par, na zmianę wykonują zagrania, które mają na celu minimalizację liczby lew (obrońcy) oraz maksymalizację liczby lew (rozgrywający).

Przykład: *Gracz W, wistując ma na celu minimalizację liczby lew wziętych przez rozgrywającego, natomiast rozgrywający, gdy dokłada kartę do pierwszego wyjścia, ma na celu maksymalizację liczby lew. Następnie, drugi obrońca znowu staje przed problemem wyboru zagrania mającego na celu minimalizację liczby lew rozgrywającego,*

12 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009.
13 Piotr Beling, *Efektywne rozwiązanie problemu rozgrywki w otwarte karty w brydżu* [w] *Metody Informatyki Stosowanej 3/2011 (28)*, Szczecin 2011, s. 5.
14 Tamże.

wreszcie rozgrywający ponownie ma za zadanie maksymalizację liczby lew, które ma wziąć.

Aby zaimplementować algorytm Min-Max dla rozgrywki brydżowej, należy zatem rozpatrywać przypadek gry dwuosobowej o stałej sumie wypłat. Piotr Beling stwierdził, że teoretycznie można tego dokonać poprzez implementację algorytmu rekurencyjnego, ale jak sam podaje: *będzie to jednak wysoce nieefektywne, zaś czas obliczeń będzie nieakceptowalny*[15]. Z tego powodu opracowano szereg ulepszeń i heurystyk, które znacznie poprawiają złożoność tego algorytmu. Między innymi są to: algorytm cięć alfa-beta i jego pochodne (MTD), wiele metod porządkowania ruchów, tablica transpozycji i iteracyjne pogłębianie. *Dla efektywnego rozwiązania problemu rozgrywki w otwarte karty, ogromne znaczenie ma także zastosowanie algorytmu Partition Search*[16].

Algorytm alfa-beta to podstawowa metoda obcinania gałęzi w drzewach minimaksowych. Jest on modyfikacją algorytmu Min-Max, która polega na tym, że dokładnych wartości minimaksowych poszukuje on tylko w zadanym przedziale wartości (alfa, beta), zwanym „oknem”[17]. Algorytm alfa-beta wykorzystuje tablicę transpozycji (albo tablicę przejść), w której zapisywane są obliczone wartości (lub górne/dolne ograniczenia tych wartości) poszczególnych stanów gry.

Algorytmy z rodziny MTD (ang. Memory-enhanced Test Driver) polegają na stopniowym zawężaniu przedziału potencjalnych wartości minimaksowych rozważanej pozycji za pomocą serii wywołań fail-soft alfa-beta z tablicą transpozycji w zerowym oknie, tj. dla takich wartości alfa, beta, że nie istnieje wartość wypłaty A, taka że alfa < A < beta (różnica między alfa i beta równa jest najmniejszej możliwej, niepodzielnej jednostce wypłaty)[18].

Idea algorytmu Partition Search polega na uogólnianiu wiedzy zdobytej podczas przeszukiwania stanu gry (a dokładnie jego wartości minimaksowej), na stany „zbliżone” do obecnie badanego. *Algorytm, oblicza nie tylko wartość minimaksową (lub jej ograniczenie) zadanej pozycji s, ale także, zbiór stanów (zawierający s), które mają tą samą wartość minimaksową (lub takie same jej ograniczenie) co s. Każda zapisywana do tablicy transpozycji informacja dotyczy więc nie pojedynczego stanu, lecz obliczonego,*

15 Piotr Beling, *Efektywne rozwiązanie problemu rozgrywki w otwarte karty w brydżu* [w] *Metody Informatyki Stosowanej 3/2011 (28)*, Szczecin 2011, s. 8.
16 Tamże.
17 Tamże, s. 9.
18 Tamże, s. 11.

niepustego zbioru stanów. (…) Umożliwia to większą liczbę trafień i odcięć w oparciu o Tablicę Transpozycji[19].

2.4. Statystyka

Populacja to zbiór wszystkich rozdań brydżowych spełniających ograniczenia ustalone na potrzeby eksperymentu.

Próba statystyczna to zbiór obserwacji statystycznych wybranych z populacji, w tym przypadku będzie to pewien wyznaczony w sposób losowy podzbiór rozdań, które spełniają określone warunki (zatem jest to tzw. **próba losowa**).

Rozmiar próby to moc wyznaczonego podzbioru obserwacji (rozdań).

Wartość oczekiwana to wartość określająca spodziewany wynik doświadczenia losowego (inaczej tzw. **pierwszy moment zwykły**).

Jeśli dyskretna zmienna losowa X przyjmuje wartości $x_{1,}x_{2,}\ldots,x_n$, z prawdopodobieństwem odpowiednio: $p_{1,}p_{2,}\ldots,p_n$, to wartość oczekiwana EX zmiennej losowej X wyraża się wzorem:

$$EX=\sum_{i=0}^{n} x_i p_i \quad .$$

Przykład: *W celu obliczenia wartości oczekiwanej dla podejmowanej decyzji, wyznaczamy wektor wyników oraz rozkład lew, a następnie obliczamy ich iloczyn skalarny. Mając dane: kontrakt 4♠ (przed partią, bez kontry) oraz rozkład lew* $R=[0,0,0,0,0,0.1,0.1,0.1,0.1,0.2,0.4,0,0,0]$ *, bierzemy wektor wyników dla kontraktu 4♠* $W(4\spadesuit)=[-500,-450,-400,-350,-300,-250,-200,-150,-100,-50,420,450,480,510]$. *Następnie obliczamy* $EX=W\circ R=93$.

Wariancja to średnia arytmetyczna kwadratów różnic poszczególnych wartości cechy od wartości oczekiwanej. Wariancja zmiennej losowej X oznaczoną jako $Var[X]$ lub $D^2(X)$, zdefiniowana jest wzorem: $Var[X]=E[X-\mu]$, gdzie $E[...]$ to wartość oczekiwana zmiennej losowej podanej w nawiasach kwadratowych, a μ jest wartością oczekiwaną zmiennej X .

Odchylenie standardowe zmiennej losowej to pierwiastek kwadratowy wariancji. Wzór na odchylenie standardowe to: $\sigma=\sqrt{(Var[X])}$.

19 Piotr Beling, *Efektywne rozwiązanie problemu rozgrywki w otwarte karty w brydżu* [w] *Metody Informatyki Stosowanej 3/2011 (28)*, Szczecin 2011, s. 11.

3. Stan prac

W tym rozdziale zostaną przedstawione rezultaty badań nad generowaniem rozdań z ograniczeniami (na podstawie pracy Pawła Perza), a także trzy programy służące do analizy rozdań brydżowych. W podrozdziale 3.2 będą omówione dwa programy służące do analizy rozgrywki w otwarte karty: DeepFinesse (DF) i wykorzystany w projekcie BridgeCalculator (BC). Natomiast w podrozdziale 3.3 zaprezentowany zostanie Analizator9000, który wykorzystuje zarówno generator rozdań z ograniczeniami, jak i program do rozgrywki (należący do pakietu BC), umożliwiając prowadzenie badań statystycznych.

3.1. Generowanie rozdań z ograniczeniami

Problem generowania rozdań z ograniczeniami został przeanalizowany w pracy *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*[20] napisanej przez Pawła Perza.

Autor pracy zaczął od opisu problemu generowania liczb pseudolosowych (opisane zostały generatory *java.util.Random* oraz *java.security.SecureRandom*), następnie uwagę poświęcił programowi bigDeal (którego działanie oparte jest na idei wygenerowania liczby losowej z odpowiedniego zakresu, a następnie odczytania rozkładu kart na podstawie tej liczby). Przy okazji wykazał, że problem braku pewności co do losowości rozdań generowanych za pomocą komputerów został już rozwiązany.

Główna część pracy Perza została poświęcona dwóm algorytmom generowania rozdań z ograniczeniami. Najpierw został omówiony algorytm typu Monte Carlo, a następnie algorytm grafowy. Efektywność tych algorytmów została porównana dla mało[21], średnio[22] i bardzo[23] skomplikowanych ograniczeń narzuconych rozdaniom. Wyniki tego eksperymentu były następujące: dla małych lub średnich ograniczeń, algorytm typu Monte Carlo był bardziej efektywny (potrzebował znacznie mniej czasu na znalezienie

20 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009.

21 Tamże, s. 51. Przykłady podane jako „proste ograniczenia”: N ma 5-10 pików / N ma 0-5 pików / N ma 10-20 punktów / N ma 5-10 pików, a S ma 5-10 kierów / N ma 5-10 pików, S ma 5-10 kierów, a E ma 5-10 trefli / N ma 5-10 pików, S ma 5-10 kierów, E ma 5-10 trefli, a W 5-10 kar. Dla dwóch ostatnich zestawów ograniczeń, algorytm grafowy działał już szybciej niż algorytm Monte Carlo.

22 Tamże, s. 52. Przykłady podane jako „średnie ograniczenia”: N ma 5-10 pików, a S 3-5 pików / N ma 5-10 pików i 12-17 punktów / N ma 10-20 punktów, a S 10-20 punktów / N ma 5-10 pików i 12-17 punktów, a S ma 5-10 pików i 7-17 punktów (w tym przypadku wyraźną przewagę miał algorytm grafowy, który był aż pięć razy szybszy) / Każdy z graczy ma 8-11 punktów.

23 Tamże, s. 53. Przykłady podane jako „bardzo rozbudowane ograniczenia”: N ma 5-10 pików, S ma 3-5 pików, E ma 3-5 pików, W ma 1-4 piki / N ma 12-20 punktów, 7 pików i 6 kar, S ma 7-9 punktów, 6 pików, 4 kara i 0 trefli, E ma 0-3 trefle.

oczekiwanej liczby rozdań spełniających zadane parametry), natomiast gdy ograniczenia były bardzo skomplikowane, przewagę miał algorytm grafowy (algorytm typu Monte Carlo w ogóle miał problemy ze znalezieniem odpowiednich rozdań[24]). Trzeba jednak zauważyć, że podział ograniczeń na mało, średnio i bardzo skomplikowane, dokonany przez autora, oparty był raczej na intuicji brydżysty, niż na rozkładach matematycznych (do których się przy tej okazji w ogóle nie odwoływał, chociaż wcześniej z nich korzystał przy weryfikacji losowości generowanych rozdań).

Autor stwierdził również, że: *Algorytm 2* [grafowy] *wypada o wiele gorzej w testach na prostych przykładach. Częściowo winą za to ponosi brak optymalizacji implementacji i dosyć duża ilość operacji na obiektach*[25]. Pojawił się także inny problem: *Algorytm 2 znajduje skomplikowane rozdania o wiele szybciej niż algorytm 1 (algorytm 1 dla skomplikowanych ograniczeń staje się zupełnie nieprzydatny ze względu na zbyt dużą złożoność czasową). Niestety algorytm 2 w podstawowej formie znajduje rozdania skrajne, tzn. takie, w których ilość punktów i skład przyjmują wartości skrajne z możliwych*[26]. Jest to dosyć poważna wada, ponieważ badania statystyczne nie powinny być prowadzone w oparciu o skrajne rozdania, a istnieje poważne ryzyko, że dla bardziej skomplikowanych ograniczeń tylko takie zostaną znalezione. Dlatego to, że: *Algorytm 2 ma dodatkową zaletę: nawet jeśli nie odnajdzie pasującego rozdania, doprowadza do rozdania, które jest zbliżone do szukanego*[27], nie wystarczyło aby stwierdzić, że na potrzeby niniejszej pracy warto implementować algorytm grafowy (w opisanej formie).

Ostatnią część pracy stanowi opis aplikacji powstałej w ramach projektu magisterskiego[28]. W podsumowaniu autor stwierdził, że oba wykorzystane przez niego algorytmy nadają się do generowania rozdań z określonymi ograniczeniami. *Tym niemniej nadal wymagają dalszych prac nad odszukiwaniem bardziej statystycznych rozdań, jak również nad poprawieniem implementacji w celu zredukowania zużycia pamięci oraz poprawienia czasu działania*[29].

Pod wieloma względami, praca Pawła Perza jest ważnym fundamentem dla niniejszej pracy, zwłaszcza jeśli chodzi o wiedzę na temat generowania rozdań z ograniczeniami. Na podstawie informacji w niej zawartych, na potrzeby projektu

24 W pracy Pawła Perza nie zostały podane bardziej szczegółowe informacje na temat dokładnej implementacji algorytmu typu Monte Carlo, więc problem ten mógł wynikać np. z braku optymalizacji.
25 Paweł Perz, *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009, s. 53.
26 Tamże.
27 Tamże, s. 54.
28 Tamże, s. 57 i dalsze.
29 Tamże, s. 61.

magisterskiego został wybrany algorytmu typu Monte Carlo. Metody sprawdzania losowości generowanych rozdań (patrz: 4.2.1), również zostały zaczerpnięte z tego opracowania. Warto zaznaczyć, że w pracy Perza rozdania były generowane w oparciu o ograniczenia wynikające z licytacji, ale nie zakładano znajomości żadnej z rąk. Już znajomość jednej ręki bardzo poważnie zmniejsza przestrzeń przeszukiwanych rozdań, w konsekwencji poprawiając efektywność algorytmu typu Monte Carlo. Gdy badane są prawa brydżowe o ogólnym charakterze, dotyczą one zazwyczaj na tyle prawdopodobnych sytuacji, że algorytm typu Monte Carlo nie ma trudności ze znalezieniem odpowiednich rozdań. W przypadku gdy rozważany jest konkretny problem decyzyjny, na ogół dysponujemy pełną informacją na temat z jednej rąk oraz zestawami ograniczeń dla trzech pozostałych, zatem generowanie rozdań odbywa się w oparciu o 39 zakrytych kart, a nie 52. W efekcie tego, „przestrzeń” przeszukiwanych rozdań już na samym początku ulega poważnej redukcji.

3.2. Programy do rozgrywki w otwarte karty

3.2.1. DeepFinesse

Jest to program który został stworzony przez Williama Bailey’a w 1993 roku. Służy do analizy rozdań z wykorzystaniem rozgrywki w otwarte karty. Przez większość czasu DeepFinesse był dostępny jedynie w wersji płatnej, ale najnowsza wersja (wydana w 2014 roku) jest w pełni darmowa (można ją pobrać z oficjalnej strony programu http://www.deepfinesse.com).

W celu dokonania analizy rozdania, użytkownik (za pomocą graficznego interfejsu) może samodzielnie wprowadzić karty należące do każdej z rąk, albo wczytać gotowy rozkład z pliku. Po wprowadzeniu rozdania, należy wybrać kontrakt oraz rękę rozgrywającego (patrz: rysunek 3.1). Rozgrywka może teraz zostać przeprowadzona ręcznie „krok po kroku”, albo z wykorzystaniem jednego z trybów automatycznych, do których należą m. in. automatyczna rozgrywka od momentu, gdy będzie już tylko jedna linia rozgrywki prowadząca do wygrania kontraktu, albo od momentu gdy dostępna jest już dowolna taka linia rozgrywki.

Rysunek 3.1

Użytkownik może analizować kolejne zagrania, na każdym etapie rozgrywki dysponując informacją o tym, które zagranie prowadzi do sukcesu (takie zagrania oznaczane są przez „W"), a które do porażki („L").. Na rysunku 3.1 S rozgrywa kontrakt 7♥. Jako wist został wybrany ♦W, który jest oznaczony przez DF jako „L", czyli zagranie tej karty przez gracza W spowoduje, że para WE nie będzie już w stanie obłożyć kontraktu. Natomiast trefle oraz piki w ręce W są oznaczone jako „W", a zatem gdyby zawistowano w dowolną z tych kart, WE ciągle mogliby uniemożliwić realizację kontraktu.

Po ustawieniu opcji „Autoplay" na automatyczne wybieranie dowolnej wygrywającej linii rozgrywki, zamiast ręcznego wybierania kolejnych kart (na podstawie podpowiedzi „W" i „L") można w prawym dolnym rogu przejść przez kolejne zagrania wykonywane przez program. Na ostateczny wynik składają się liczby lew wziętych przez pary NS i EW.

DF od kilkunastu lat w społeczności brydżystów jest uznawany za „autorytet", jeśli chodzi o kwestie związane z techniką rozgrywki[30]. Przyjmuje się, że jeżeli DF nie jest w

30 *Deep Finesse, program Williama Baileya, dokonujący szybkiej analizy rozgrywki w otwarte karty, stał się dla wielu brydżystów wyrocznią w (czasami niebanalnych) sporach na temat, czy dany kontrakt idzie. Wszyscy też przyzwyczailiśmy się do wyliczanych za jego pomocą i drukowanych wraz z rozkładami tabelek z możliwą do*

stanie znaleźć wygrywającej linii rozgrywki, to taka linia nie istnieje. W programie analiza rozgrywki wykonywana jest za pomocą implementacji algorytmu Min-Max (opisanego w podrozdziale 2.3). Nieomylność wersji wydanej w 1999 została podważona, o czym na swojej stronie internetowej napisał Piotr Beling: *Duża ilość usprawnień* [zastosowanych w algorytmie Min-Max – przyp. autora] *znacznie przyspiesza obliczenia, ale także komplikuje kod źródłowy programu. Zwiększa szansę wystąpienia w nim pomyłek, a co za tym idzie złych wyników podawanych przez program. Niekiedy błędy ujawniają się na tyle rzadko, że potrzeba wielu lat by je odkryć. Jeden, po prawie 10 latach od wydania ostatniej wersji programu, został znaleziony w samym Deep Finesse w trakcie czwartych Otwartych Mistrzostw Europy w San Remo w 2009 roku*[31].

Na oficjalnej stronie programu nie są dostępne żadne informacje na temat możliwości wykonywania analizy z pominięciem interfejsu graficznego, oraz automatycznego zapisu tak uzyskanych wyników do pliku. Z tego powodu DF nie został wykorzystany w projekcie.

3.2.2. BridgeCalculator

Piotr Beling[32] na stronie bcalc.w8.pl udostępnia pakiet darmowych narzędzi, które służą do wykonywania różnych obliczeń związanych z grą w brydża. Najnowsza wersja 15.12.1 została udostępniona 19 grudnia 2015. W skład pakietu wchodzą trzy programy: wyposażony w graficzny interfejs program rozwiązujący problem rozgrywki w otwarte i zakryte karty, program do generowania rozdań zgodnych z zadanymi ograniczeniami (z tekstowym interfejsem) oraz konsolowa wersja programu rozwiązującego problem rozgrywki w otwarte karty. Ten ostatni został wykorzystany w projekcie stanowiącym część niniejszej pracy, co było możliwe dzięki obszernej i przejrzystej dokumentacji[33] (sposób jego wykorzystania zostanie opisany w podrozdziale 4.1.2.). Ponieważ program do generowania rozdań z ograniczeniami został już opisany w (3.1.), szerzej omówiony zostanie jedynie program do rozwiązywania problemu rozgrywki w otwarte karty (w wersji z graficznym interfejsem). W programie tym wykorzystana została implementacja algorytmu Min-Max w języku C++.

---

wzięcia ilością lew. - napisał Piotr Beling w artykule pt. *Deep Finesse nieomylny, ale nie do końca* dostępnym na stronie: http://qwak.w8.pl/bridge:deep_finesse_omylny

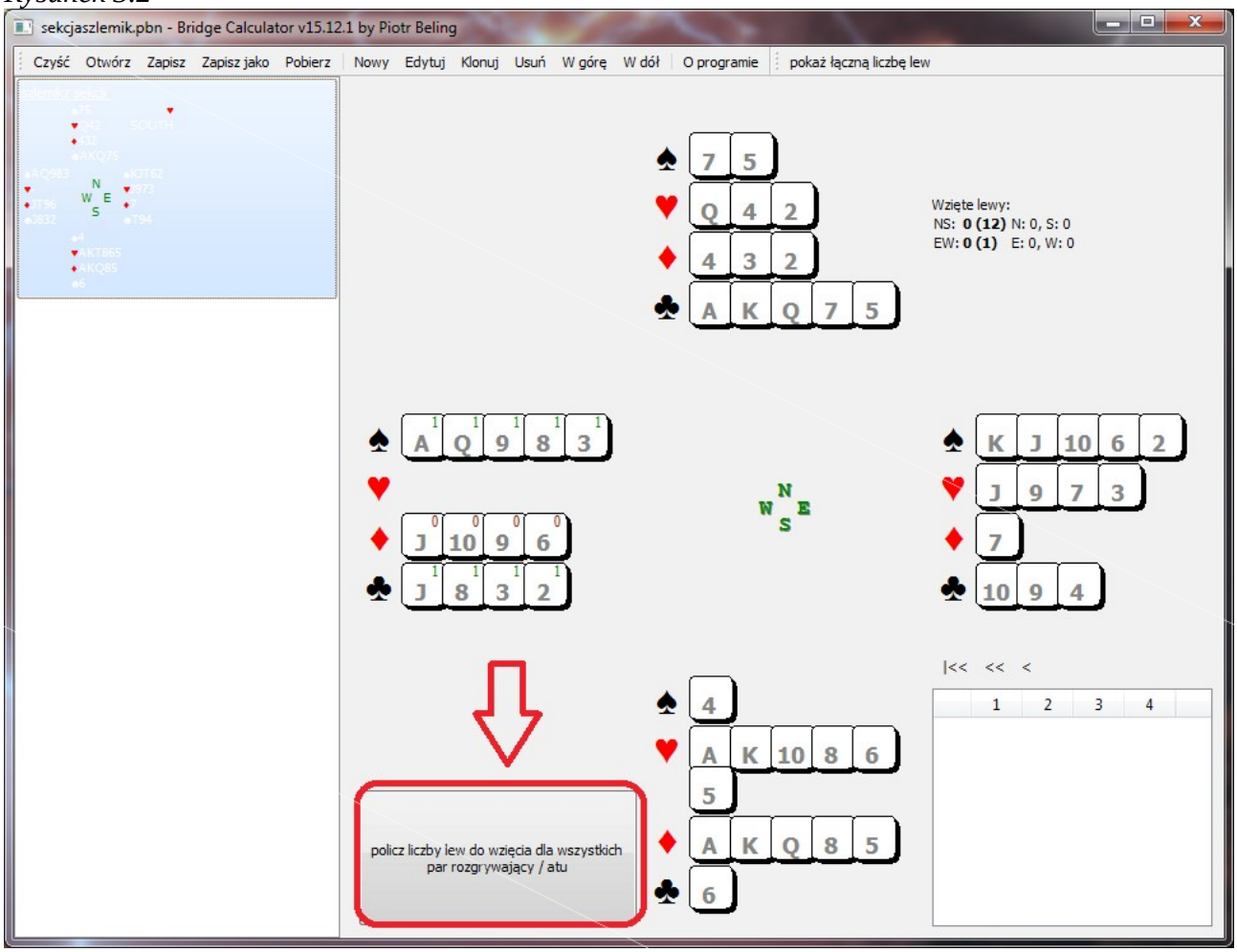
31 Piotr Beling, *Deep Finesse nieomylny, ale nie do końca,* Źródło*:* http://qwak.w8.pl/bridge:deep_finesse_omylny

32 Piotr Beling – informatyk, w 2006 obronił pracę magisterską pt. *Praktyczne aspekty programowania gier logicznych* na Politechnice Łódzkiej. Jak sam podaje: *BridgeCalculator rozwijany jest w ramach hobby, z nadzieją, że będzie użyteczny. Ponadto, jego silnik projektowany jest na potrzeby pracy naukowej (doktoratu) której celem jest zbadanie różnych algorytmów sztucznej inteligencji pozwalających tworzyć programy grające w gry logiczne z niedoskonałą informacją (na przykładzie gry w brydża).* Źródło: bcalc.w8.pl .

33 Dokumentacja dostępna na stronie: http://bcalc.w8.pl/index.php?lang=pl&topic=help .

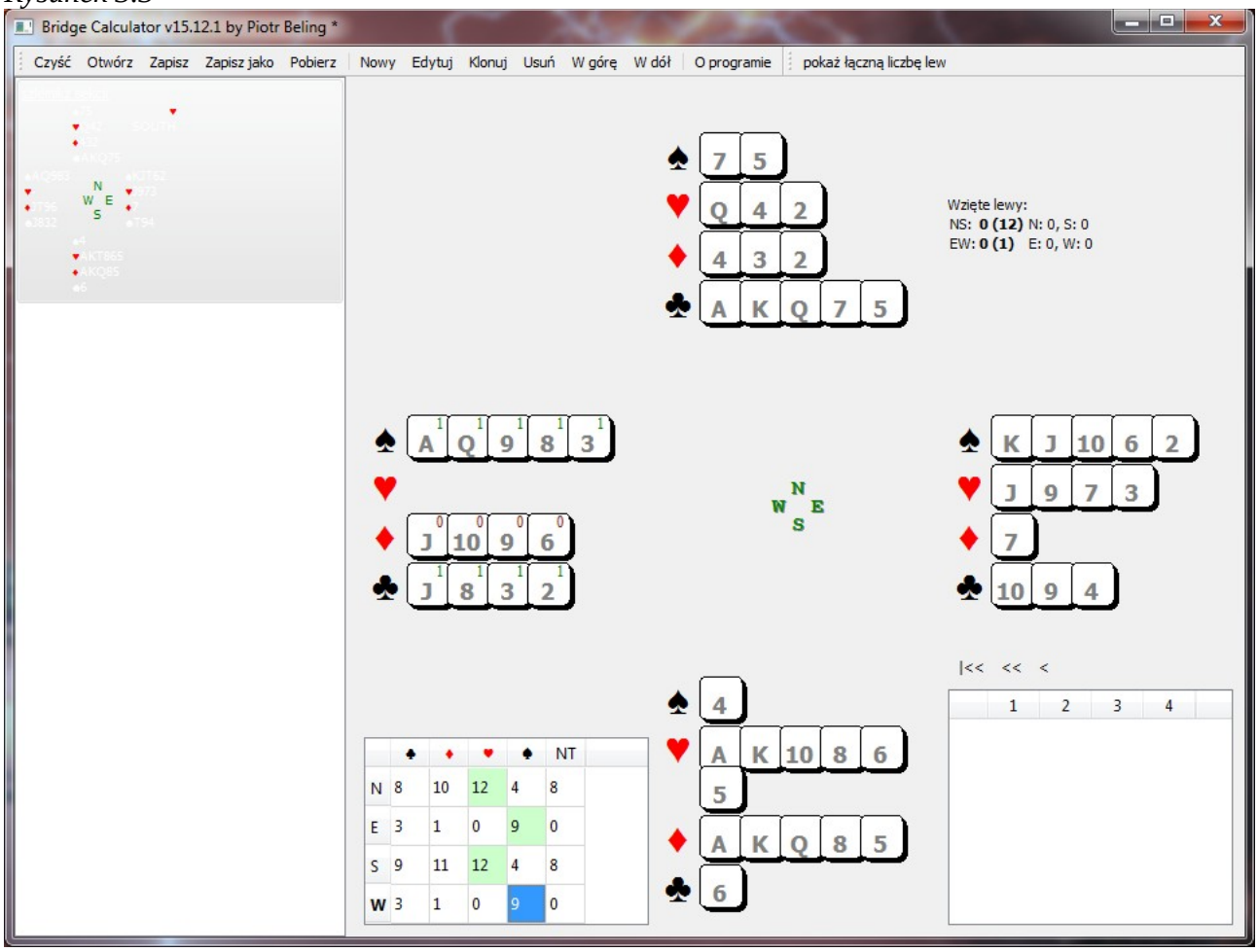
Aby dokonać analizy rozdania, użytkownik zaczyna od wyboru jednej z trzech opcji: „Otwórz” (otwiera rozdanie z pliku zapisanego na dysku), „Pobierz” (użytkownik podaje lokalizację z której zostaje pobrany plik z rozdaniem) lub „Nowy”. W przypadku wybrania tej ostatniej opcji, użytkownik musi ustalić rodzaj problemu – do wyboru jest rozgrywka w otwarte karty, oraz gdy dwie ręce są znane, a dwie zakryte (przypadki te zostaną opisane osobno).

Rysunek 3.2



Po wybraniu rodzaju problemu „w otwarte karty” użytkownik wprowadza rozkład kart, wpisując je ręcznie w ustalonym formacie (np. AK543.K87.A2.KT to układ ♠AK543 ♥K87 ♦A2 ♣K10), co jest mniej wygodnym rozwiązaniem niż zastosowane w DF. Należy również dokonać wyboru koloru atutowego oraz ręki rozgrywającego (lub wistującego), a następnie zatwierdzić wprowadzone parametry. Po zatwierdzeniu, rozdanie pojawi się po lewej stronie interfejsu. Aby do niego przejść, należy je wybrać za pomocą kliknięcia. Po wejściu w rozdanie, użytkownik może wykonywać rozgrywkę „krok po kroku”, w sposób analogiczny do DF, albo wybrać duży przycisk dostępny na dole interfejsu i za jego pomocą od razu wyznaczyć rozkład lew w tym rozdaniu (patrz rysunek 3.2).

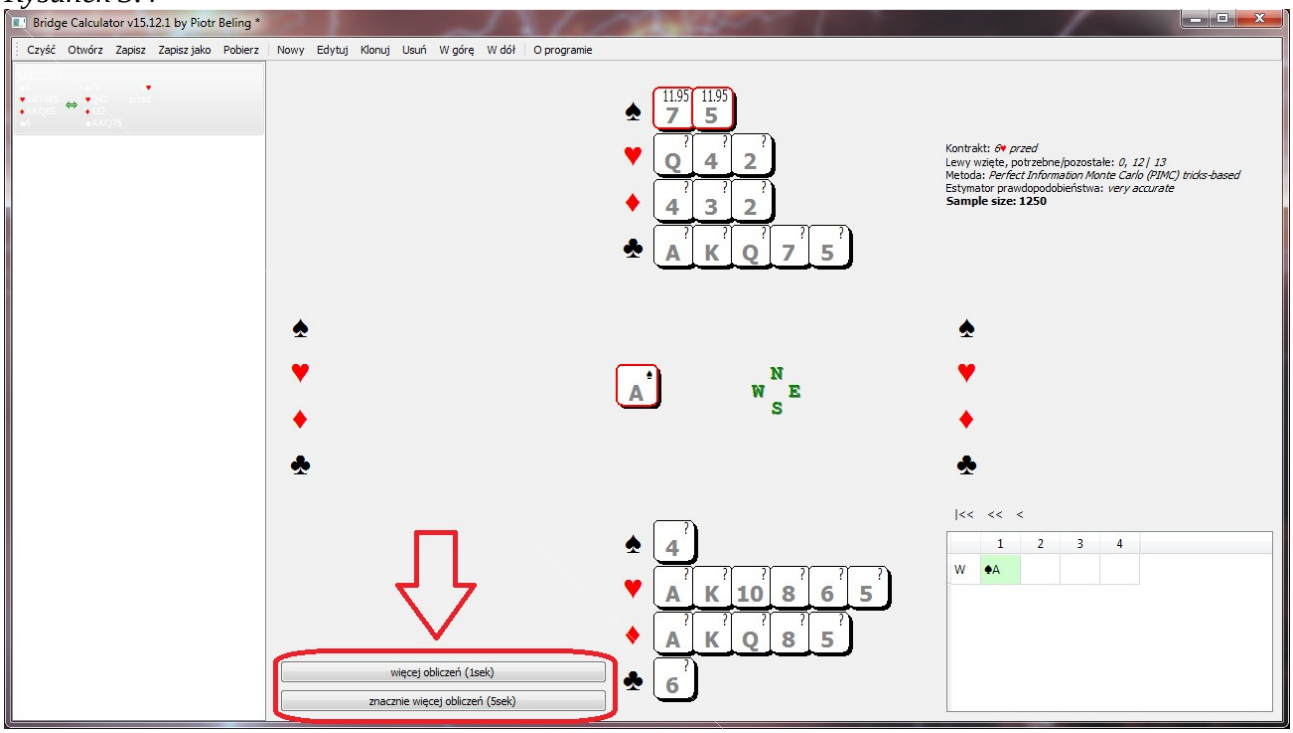
Rysunek 3.3



Na rysunku 3.3, zielonym tłem zostały zaznaczone maksymalne liczby lew dla każdej pary (dla NS jest to 12 lew w kiery, dla WE 9 lew w piki). W przypadku, gdy użytkownik zdecyduje się na samodzielną analizę rozgrywki, obliczenia wykonywane przez program w tle będą praktycznie niezauważalne. W takim przypadku pomocne będą zielone liczby pojawiające się na górze każdej z kart dostępnych do zagrania (gdy zagranie jest korzystne) oraz czerwone (gdy zagranie jest niekorzystne).

Jeśli wybrana zostanie opcja analizy rozgrywki przy zakrytych kartach, użytkownik będzie musiał wprowadzić ręce graczy S (rozgrywającego) i N (dziadka), a także te karty obrońców, które są już znane (w przypadku gdy rozgrywka dopiero się rozpoczyna, jedyną znaną kartą będzie karta wistu należąca do W). Na potrzeby obliczeń trzeba też wybrać jedną z dostępnych metod np. Perfect Information Monte Carlo (z dokumentacji nie wynika, czy chodzi tu o sam algorytm generowania rozdań czy może o jakąś uproszczoną wersję algorytmu Min-Max) oraz estymator prawdopodobieństwa (trzy możliwości, od bardzo szybkiego do bardzo dokładnego). Po zatwierdzeniu tych ustawień, pojawia się widok jak na rysunku 3.4.

Rysunek 3.4



W tym przypadku ustalenie rozkładu lew nie jest możliwe – tak jak w sytuacji gdy cały rozkład kart jest znany, możliwa jest jedynie ręczna analiza rozgrywki. W tym celu użytkownik może wybrać jedną kartę (np. kartę wistu) z puli kart należących do obrońców. Następnie program wykonuje obliczenia i na ich podstawie informuje użytkownika, która z dostępnych możliwości jest najbardziej korzystna dla rozgrywającego. W przypadku przedstawionym na rysunku 3.4, gracz W zawistował ♠A, natomiast program obliczył, że między zagraniem z dziadka ♠5 i ♠7 nie ma widocznej różnicy (wybór dowolnej z tych kart daje średnio 11,95 lewy – liczby te są podane na górze dostępnych kart). W ten sposób, nawet bez pełnej znajomości rąk przeciwników można ustalić, które zagranie jest w danej sytuacji najbardziej opłacalne.

Gdy analizowane są problemy rozgrywki przy zakrytych kartach przeciwników, program działa wyraźnie wolniej, ponieważ w tym przypadku zachodzi konieczność rozgrywania wielu rozdań. Poniżej rozdania widoczne są przyciski, które pozwalają na zwiększenie liczby rozdań wykorzystanych do estymacji (1000 dodatkowych rozdań „kosztuje” około 5 sekund).

3.3. Analizator9000

W 2012 roku Michał Kilichowicz na stronie an9k.emkael.info udostępnił stworzony przez siebie program Analizator9000. Projekt scharakteryzował w następujący sposób: *Analizator9000 jest aplikacją oferującą graficzny interfejs użytkownika dla analizy*

statystycznej liczby lew w rozdaniach wygenerowanych na podstawie określonych warunków[34]. W ramach tej aplikacji zostały wykorzystane dwa inne, samodzielne programy: Dealer[35] (do generowania rozdań spełniających określone warunki), oraz konsolowa wersja opisanego w podrozdziale 3.2.2 programu BC (do rozwiązywania problemu rozgrywki w otwarte karty). Program do uruchomienia wymaga środowiska .NET 3.5, ponieważ został napisany w języku C#.

Analizator9000 oferuje trzy funkcjonalności. Pierwszą z nich jest generowanie rozdań spełniających zadane ograniczenia, drugą – analiza liczby lew, a trzecią – analiza kontraktów.

Rozdania można wygenerować na podstawie znanych kart (informacji o tym, w której ręce znajduje się konkretna karta), oraz zdefiniowanych warunków. Można również zdefiniować określone zdarzenia, które użytkownik chce zbadać, np. jak często przy zadanych ograniczeniach, liczba kierów w ręce S będzie większa od liczby trefli w ręce S. Przy definiowaniu ograniczeń dla generowanych rozkładów, użytkownik może stosować koniunkcję („&&”), alternatywę („||”), negację („!”) oraz operatory: „>”, „<”, „<=”, „>=”, „==”. Format ten jest zgodny z wymaganym przez program Dealer (warunki wpisane przez użytkownika w interfejsie Analizator9000 przekazywane są do programu Dealer). Jeśli użytkownik ma własny plik z rozdaniami, może je wczytać zamiast generowania.

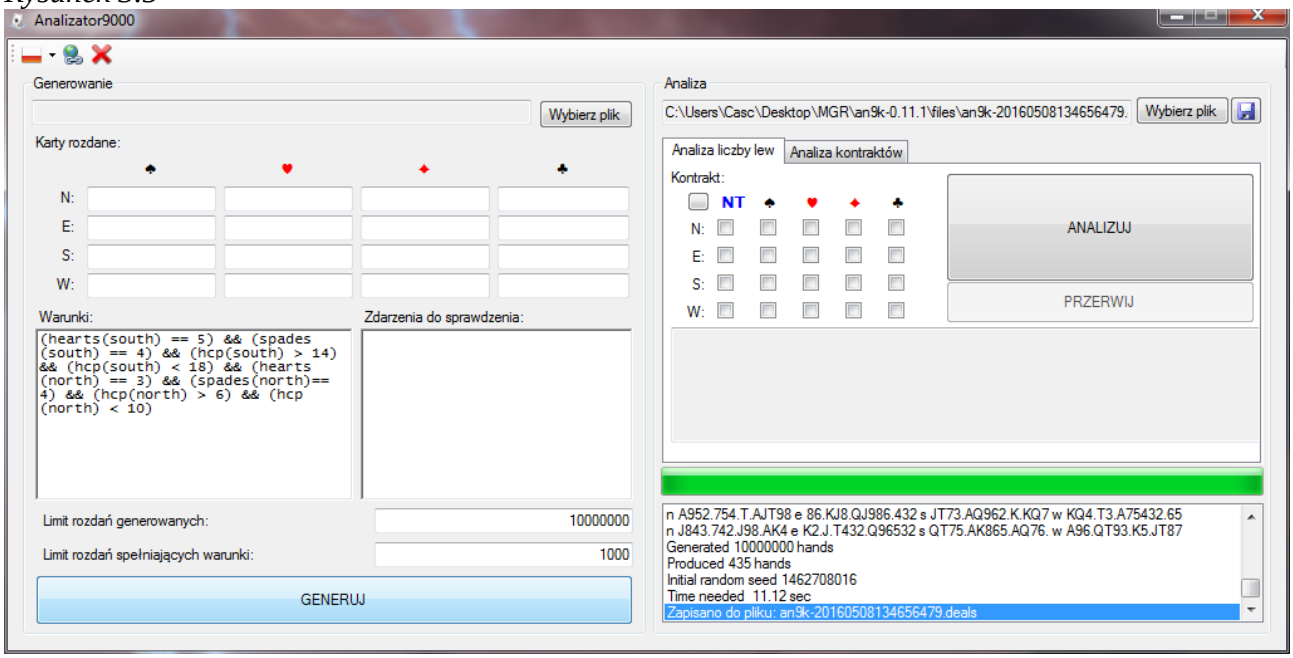
Przykładowo: aby wygenerować takie rozdania, w których S ma od 15 do 17 punktów oraz dokładnie 5 kierów i dokładnie 4 piki, natomiast N ma od 7 do 9 punktów i dokładnie 3 kiery oraz dokładnie 4 piki, do pola definiowania ograniczeń należy wpisać: **(hearts(south) == 5) && (spades(south) == 4) && (hcp(south) > 14) && (hcp(south) <**

34 Michał Kilichowicz, opis z strony: an9k.emkael.info

35 Program Dealer dostępny był na stronie henku.home.xs4all.nl/html/dealer/software.html, obecnie ta podstrona nie istnieje. Na stronie z opisem henku.home.xs4all.nl/html/dealer/dealer.html można przeczytać, że: *The program dealer can be used to generate hands for partnership bidding training, or for generating statistics, that can be used to design conventions or win postmortems*, w wolnym tłumaczeniu: *Program dealer można wykorzystać do generowania rąk do trenowania licytacji w parze, do generowania statystyk, które mogą zostać wykorzystane do tworzenia konwencji lub do wygrywania sekcji zwłok*. Ten ostatni przypadek może brzmieć nieco egzotycznie: w języku potocznym brydżystów, chodzi tu o sytuację w której jeden z partnerów (lub z członków drużyny) wykazuje innemu partnerowi (członkowi drużyny), że decyzja przez niego podjęta była słuszna (z punktu widzenia statystyki), natomiast w tym konkretnym rozdaniu wystąpił wyjątkowo niekorzystny rozkład kart, który spowodował słaby zapis dla pary (drużyny). Dealer powstał w połowie lat ‘90, a jego autorem jest Hans van Staveren, natomiast od końca lat ‘90 projektem opiekuje się Henk Uijterwaal. Z dokumentacji dostępnej na stronie projektu nie udało się ustalić, jaki algorytm został wykorzystany do generowania rozdań z ograniczeniami w Dealerze, pojawia się tylko informacja, że: *The speed of the program obviously depends on the machine it runs on. On Han's home machine, a SparcStation 1+ it generates about 4000 hands a second if the condition is not too complicated*, w wolnym tłumaczeniu: *Szybkość programu oczywiście zależy od urządzenia, na którym jest uruchamiany. Na domowym sprzęcie Hansa, czyli na SparcStation 1+, generowanych jest około 4000 rozdań na sekundę, o ile warunki nie są zbyt skomplikowane*. Na podstawie tych informacji, oraz wyników badań Pawła Perza (przedstawionych w podrozdziale 3.1) można jedynie przypuszczać, że w Dealerze nie został zastosowany algorytm grafowy (którego czas działania w znacznie mniejszym stopniu zależy od poziomu komplikacji ograniczeń generowanych rozdań).

18) && hearts(north) == 3) && (spades(north) == 4) && (hcp(north) > 6) && (hcp(north) < 10). Zanim użytkownik rozpocznie generowanie, może jeszcze zmienić (ponieważ znajdują się w nich wartości domyślne) zawartość dwóch pól: „Liczba rozdań generowanych” (czyli ile rozdań generator ma przejrzeć w poszukiwaniu rozdań spełniających zdefiniowane warunki), oraz „Liczba rozdań spełniających warunki” (po której osiągnięciu generator zakończy działanie). Gdy użytkownik ustawi wszystkie parametry, za pomocą przycisku „Generuj” uruchamia generowanie rozdań i następnie czeka na informację zwrotną, która pojawia się w prawym dolnym rogu programu (rysunek 3.5). Wygenerowane rozdania automatycznie zostają zapisane do pliku.

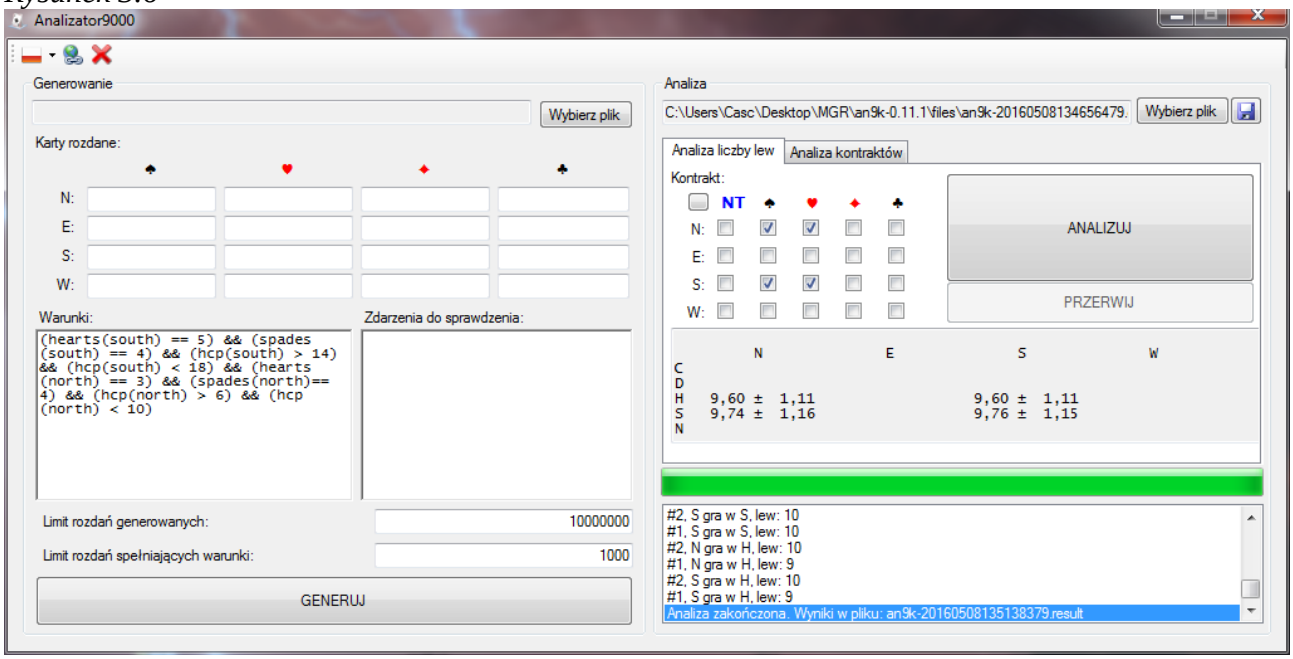
Rysunek 3.5



Na rysunku 3.5 widać, że pomimo tego, iż program sprawdził aż 10.000.000 rozdań, nie udało mu się znaleźć 1000 rozdań spełniających zdefiniowane warunki. Znalazł ich jedynie 435, co zajęło mu 11,12 sekundy.

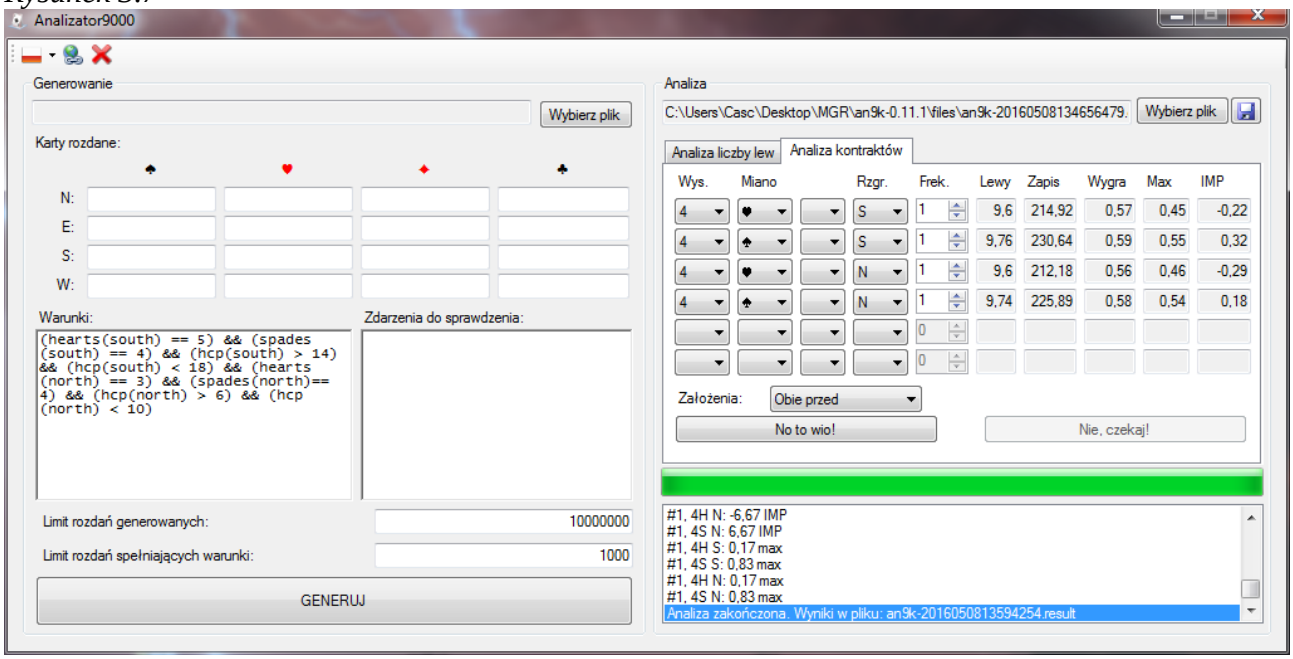
Po wygenerowaniu (lub wczytaniu) rozdań, użytkownik może przejść do analizy. Jeśli zdecyduje się na analizę liczby lew (która jest włączona domyślnie), to zaznacza kolory atutowe (lub BA) oraz pozycję rozgrywającego; może wybrać jedną możliwość, kilka, albo nawet wszystkie (opcja ta jest dostępna w lewym górnym rogu panelu). Przykładowo, dla wygenerowanych już rozdań, użytkownik chce sprawdzić liczbę lew do wzięcia w kiery oraz w piki, z rąk S i N. W tym celu zaznacza cztery kratki (patrz: rysunek 3.6), a następnie klika w przycisk „Analizuj”. Gdy analiza zostanie przeprowadzona, pod panelem wyboru kontraktów i rąk rozgrywającego zostaną wyświetlone rezultaty (jednocześnie zostaną też zapisane do pliku).

Rysunek 3.6



W przypadku gdy użytkownik będzie zainteresowany analizą kontraktów, powinien przejść do odpowiedniej zakładki, a następnie ustawić interesujące go kontrakty (maksymalnie sześć), podając ich wysokość oraz kolor, informację o kontrze (rekontrze) oraz rękę rozgrywającego. Na przykład, w oparciu o wygenerowane uprzednio rozdania użytkownik może zechcieć sprawdzić wyniki dla kontraktów: 4♠ z rąk S i N oraz 4♥ z rąk S i N. Poza kontraktami, należy także wybrać założenia (na rysunku 3.7 widać, że zostały wybrane „obie przed"). Po wprowadzeniu tych ustawień, użytkownik klika w przycisk „No to wio!" i program rozpoczyna analizę. Po przeprowadzeniu analizy, jej rezultaty zostaną zapisane do pliku. Rysunek 3.7 przedstawia wyniki przykładowej analizy kontraktów.

Rysunek 3.7



Z rysunku 3.7 wynika, że według programu najbardziej opłacalnym kontraktem jest 4♠ rozgrywane z ręki S, następnie ten sam kontrakt grany z ręki N. Po prawej stronie od każdego z wprowadzonych kontraktów, pojawiły się informacje na temat średniej liczby lew („Lewy”), średniego uzyskanego zapisu („Zapis”), prawdopodobieństwa realizacji kontraktu („Wygra”), prawdopodobnego rezultatu w turnieju na zapis maksymalny („Max”) oraz w meczu, lub turnieju granym na zapis meczowy („IMP”).

Analizator9000 to program, który można z powodzeniem wykorzystywać do przeprowadzania analizy rozdań, jednak nie udostępnia on możliwości zapisywania i odczytywania gotowych eksperymentów, a także nie jest dostępny w wersji konsolowej, która umożliwiałaby automatyzację niektórych działań. Nie istnieje również możliwość generowania rozdań „do skutku”, czyli aż do momentu, gdy wygenerowana zostanie liczba rozdań podana przez użytkownika.

4. Opis rozwiązania

Rozdział ten został podzielony na dwie części. Podrozdział 4.1 poświęcono opisowi projektu magisterskiego, czyli zastosowanego generatora rozdań oraz sposobu, w jaki został wykorzystany program BridgeCalculator, napisany przez Piotra Belinga[36]. W podrozdziale 4.2. zostały zaprezentowane przykładowe problemy, które można rozwiązywać za pomocą zaprogramowanych narzędzi.

4.1. Opis techniczny projektu

4.1.1. Generatory rozdań

W ramach projektu powstało osiem generatorów, które działają na podobnych zasadach, jednak różnią się w zależności od stopnia komplikacji zadania, jakie zostało przed nimi postawione. Z tego powodu z klasy nadrzędnej *Experiment* dziedziczy osiem klas o nazwach *Type_Zero*, *Type_One* i tak dalej, aż do *Type_Seven*. Każda z nich wykorzystuje inny generator rozdań. W tabeli 4.1 zostały opisane rodzaje problemów, dla których są dedykowane poszczególne generatory i odpowiadające im podklasy eksperymentów. Do generowania rozdań w ramach eksperymentów *Type_Three* powstał także dodatkowy generator działający na nieco innych zasadach, który również zostanie opisany.

Tabela 4.1

Nazwa klasy	Zastosowanie
Type_Zero	Za pomocą tego generatora można uzyskać rozdanie bez żadnych zadanych ograniczeń.[37]
Type_One	Za pomocą tego generatora można uzyskać rozdanie, w którym znana jest ręka jednego z graczy oraz ograniczenia układu i siły dla ręki dowolnego z pozostałych graczy.
Type_Two	Za pomocą tego generatora można uzyskać rozdanie, w którym znana jest ręka jednego z graczy oraz ograniczenia układu i siły dla dowolnych dwóch rąk pozostałych graczy.
Type_Three	Za pomocą tego generatora można uzyskać rozdanie, w którym znana jest ręka jednego z graczy oraz ograniczenia układu i siły dla trzech rąk pozostałych graczy.

36 Za pomocą tego programu uzyskiwany jest rozkład lew dla każdego rozdania.
37 Więcej na temat problemu losowości generowanych rozdań w podrozdziale 4.2.1.

Nazwa klasy	Zastosowanie
Type_Four	Za pomocą tego generatora można uzyskać rozdanie, w którym ustalone są ograniczenia układu i siły dla rąk wszystkich czterech graczy.
Type_Five	Za pomocą tego generatora można uzyskać rozdanie, w którym znane są ręce dwóch dowolnych graczy.
Type_Six	Za pomocą tego generatora można uzyskać rozdanie, w którym znane są ręce dwóch dowolnych graczy oraz ograniczenia układu i siły dla ręki dowolnego z pozostałych graczy.
Type_Seven	Za pomocą tego generatora można uzyskać rozdanie, w którym znane są ręce dwóch dowolnych graczy, oraz ograniczenia układu i siły dla rąk dwóch pozostałych graczy.

Każda klasa eksperymentu ma inną funkcję generującą rozdania. Dwie z nich (*Type_Zero*, *Type_Five*) są bardzo proste:

Generator Type_Zero – pseudokod

Wejście: brak

```
1  V ← [1, 2…52]
2  shuffle(V)
3  S ← V[0:12]
4  W ← V[13:25]
5  N ← V[26:38]
6  E ← V[39:51]
7  Return [S, W, N, E]
```

Generator Type Five – pseudokod

Wejście: H1, H2

```
1  V ← [1, 2…52]
2  V ← V / (H1 + H2)
3  shuffle(V)
4  A ← V[0:12]
5  B ← V[13:25]
7  Return [H1, H2, A, B][38]
```

Pozostałe funkcje są bardziej skomplikowane i korzystają z nawrotów. Ponieważ generatory dla eksperymentów *Type_One* i *Type_Two* są uproszczonymi wersjami

38 Po wyjściu z generatora ręce poszczególnych graczy zostają dopasowane do parametrów ustawionych dla danego eksperymentu (ręce H1 i H2 mogły należeć do graczy N i S, ale także do np. S i W).

generatora *Type_Three*, natomiast *Type_Six* jest uproszczoną wersją *Type_Seven*, w całości przedstawione zostaną tylko pseudokody generatorów *Type_Three*, *Type_Four* i *Type_Seven*.

Tabela 4.2

Nazwa funkcji	Opis
check_rules_set()	Funkcja sprawdzająca, czy dany ręka spełnia podane warunki.
check_all_variants()	Funkcja, która wywołuje check_rules_set dla każdego wariantu z zestawu podanych ograniczeń.
get_random_hand()	Funkcja, która otrzymuje na wejściu listę reprezentującą talię kart (lub jej część) i zwraca losowy podzbiór złożony z 13 kart.

W tabeli 4.2 opisane zostały funkcje pomocnicze, które wykorzystano w generatorach. Każdy z generatorów (poza *Type_Zero* i *Type_Five*) używa także funkcji *generate_hand*, dla której argumentami są: lista kart, z których będą generowane układy (D), zestaw ograniczeń, które muszą zostać spełnione (R), oraz liczba prób (T)[39].

Funkcja generate_hand – pseudokod

Wejście: D, R, T

```
1    for i = 0 to T
2       H ← get_random_hand(D)
3       For  j in R:
4          if check_rules_set( H, j )
5             Return H, True
6    Return [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], False
```
[40]

Generator typu trzeciego wykorzystuje nawroty i działa w oparciu o zastosowanie funkcji *generate_hand*. Wejściem do tego generatora są: ręka jednego z graczy (H), trzy zestawy ograniczeń dla rąk pozostałych graczy (W, N, E), oraz liczba prób, po których algorytm ma zakończyć działanie niezależnie od rezultatu (T).

Generator Type_Three – pseudokod

Wejście: H, N, W, E, T

39 Ponieważ wykorzystywany jest algorytm działający w oparciu o metodę Monte Carlo.

40 Jeśli po ustalonej liczbie prób (T) ani razu nie uda się uzyskać zestawu kart pasującego do któregokolwiek wariantu ograniczeń, zwrócona zostanie wartość zastępcza, która dla generatora będzie sygnałem do rozpoczęcia kolejnego przebiegu pętli.

```
1   V ← [ 1, 2 … 52 ]
2   V ← V - H
3   END ← 0
4   while END <= T
5     NH, correct ← generate_hand( V, N, T )
6     if correct:
7       D ← V - NH
8       WH, correct ← generate_hand( D, W, T )
9       if correct:
10        EH ← D - WH
11        if check_all_variants( EH, E)[41]
12          Return [ H, WH, NH, EH ], True
13        else END ++
14      else END ++
15    else END ++
16  Return generate_deal_type_0(), False[42]
```

Pseudokod generatora *Type_Two* różni się od generatora *Type_Three* tym, że zostają pominięte wiersze 11 oraz 13. W przypadku generatora *Type_One*, w wierszu ósmym zostanie wykorzystana funkcja *get_random_hand* (opisana w Tabeli 4.2) z argumentem D, a pominięte zostaną wiersze: 9, 11, 13 i 14. Oznacza to, że w praktyce generatory *Type_One* i *Type_Two* można zastąpić wywołaniem generatora *Type_Three* z odpowiednio podanymi argumentami. Wtedy jako ograniczenia dla generowanych rąk pozostałych graczy (*Type_One*) czy pozostałego gracza (*Type_Two*) można podać: [[[0,40], [0, 13], [0, 13], [0, 13], [0, 13]]], oznaczające dowolną rękę.

Generator *Type_Four* różni się od *Type_Three* jedynie tym, że na wejściu zostają podane zestawy ograniczeń dla każdej z czterech rąk.

Generator Type Four – pseudokod

Wejście: S, N, W, E, T

```
1   V ← [ 1, 2 … 52 ]
```

41 Ponieważ w talii zostało już tylko 13 kart, wywoływanie funkcji generate_hand() nie ma sensu, jednak należy sprawdzić, czy te 13 kart pasuje do zestawu ograniczeń dla ręki E, stąd konieczne jest wywołanie funkcji check_all_variants(), która w pozostałych przypadkach jest wywoływana wewnątrz generate_hand().

42 Jeżeli działanie pętli zostanie zakończone, zwrócone zostanie losowe rozdanie i wartość False, dzięki której zostanie ono zidentyfikowane jako nieudana próba wygenerowania rozdania o zadanych parametrach.

```
2    END ← 0
3    while END <= T
4       SH, correct ← generate_hand( V, S, T )
5       if correct:
6          D ← V - SH
7          NH, correct ← generate_hand( D, N, T )
8          if correct:
9             Q ← D - NH
10            WH, correct ← generate_hand( Q, W, T )
11            if correct:
12               EH ← Q - WH
13               if check_all_variants( EH, E):
13                  Return [ SH, WH, NH, EH ], True
14               else END ++
15            else END ++
16         else END ++
17      else END ++
18   Return generate_deal_type_0(), False
```

Generator *Type_Seven* działa w sposób analogiczny do *Type_Three*, ale zaczyna od usunięcia z talii dwóch podawanych na wejściu rąk.

Generator Type_Seven – pseudokod

Wejście: H1, H2, W, E, T

```
1    V ← [ 1, 2 … 52 ]
2    V ← V – (H1 + H2)
3    END ← 0
4    while END <= T
5       H3, correct ← generate_hand( V, W, T )
6       if correct:
7          H4 ← V - H3
8          if check_all_variants(H4, E):
9             Return [ H1, H2, H3, H4 ], True
10         else END ++
11      else END ++
12   Return generate_deal_type_5(H1, H2), False
```

Generator *Type_Six* różni się od *Type_Seven* tym, że pominięte zostają linie 8 i 10. Generator *Type_Seven* można stosować zamiast *Type_Six*, w sposób analogiczny do opisanego przy generatorze *Type_Three*.

Alternatywny generator Type_Three – pseudokod

Wejście: H, N, W, E

```
1   V ← [ 1, 2 … 52 ]
2   V ← V - H
3   A ← 3
4   B ← [True, False, False, False]
5   while True:
6     APART ← get_partition(V, A)
7     for i in APART:
8       if B[1] == False && check_all_variants(i, N)
9         BHAND ← i
10        B[1] ← True
11        V ← V - BHAND
12        A --
13      if B[2] == False && check_all_variants(i, W)
14        CHAND ← i
15        B[2] ← True
16        V ← V - CHAND
17        A --
18      if B[3] == False && check_all_variants(i, E)
19        DHAND ← i
20        B[1] ← True
21        V ← V - DHAND
22        A --
23      else if A == 1:
24         V ← [ 1, 2 … 52 ]
25         V ← V - H
26         A ← 3
27         B ← [True, False, False, False]
28    if A == 0:
29      return [H, CHAND, BHAND, DHAND], True
```

Funkcja get_partition – pseudokod

Wejście: V, A

```
1   partition ← []
2   for i in range (0, A):
3       X ← get_random_hand(V)
4       V ← V - X
5       partition.append(X)
6   return partition
```

Alternatywny generator rozdań typu trzeciego został oparty na nieco innej zasadzie. Otrzymuje on na wejściu rękę jednego z graczy (domyślnie S), oraz trzy zestawy ograniczeń dla pozostałych rąk. Następnie losuje podział „wolnych" kart (czyli tych, które nie należą do ręki S) i po kolei próbuje dopasować ręce z uzyskanego podziału do poszczególnych ograniczeń. W sytuacji gdy dopasuje jedną rękę, przerywa działanie wewnętrznej pętli i szuka podziału, do którego będzie w stanie dopasować dwie pozostałe ręce. W przypadku gdy zostanie tylko jedna ręka do dopasowania, generator cofnie się do stanu początkowego. Algorytm ten był testowany jako prototyp; jego ewidentną zaletą jest to, że dla poprawnie zdefiniowanych ograniczeń zawsze będzie w stanie znaleźć pasujące rozdanie, natomiast wadą to, że trudno przewidzieć czas jego działania. Wymaga on również dalszej optymalizacji.

4.1.2. Zastosowanie programu BridgeCalculator

W projekcie wykorzystany został program BridgeCalculator napisany przez Piotra Belinga, udostępniony przez autora na stronie http://bcalc.w8.pl/ do swobodnego (niekomercyjnego) użytku. Na wejściu otrzymuje on rozdanie w formacie pbn[43] oraz zbiór opcji, z których ma skorzystać w trakcie analizy rozgrywki. Na wyjściu zwracany jest rozkład lew dla podanego rozdania.

Przed uruchomieniem programu BC, rozdanie zostaje skonwertowane do formatu pbn za pomocą funkcji *pbn_board([s, w, n, e])*, w której argumentem jest rozkład kart.

43 Nazwa formatu pbn pochodzi od Portable Bridge Notation (skrót PBN). Jest to uniwersalna notacja stosowana do zapisu rozdań brydżowych, używana w programach brydżowych, którą można wykorzystywać na dowolnej platformie.

Funkcja ta zwraca zmienną typu string[44], która zostaje następnie przekazana do *beling_engine*, w której wywoływane jest następujące polecenie:

os.system('bcalconsole -e e -q -t a -d PBN -c '+pbn_board+' > trickboard.txt')

W tabeli 4.3 zostały wyjaśnione poszczególne części tego polecenia, natomiast w tabeli 4.4 zostały opisane opcje wykorzystane przy uruchamianiu *bcalconsole*.

Tabela 4.3

os.system(...)	Przekazanie do systemu operacyjnego komendy w formie zmiennej typu string
bcalconsole [argumenty]	Uruchomienie wersji konsolowej programu BridgeCalculator z określonymi argumentami (patrz: tabela 4.4)
pbn_board	Rozdanie w formacie pbn
> trickboard.txt	Przekierowanie wyjścia do pliku tekstowego

Tabela 4.4

-e e	Program zakończy działanie od razu po uruchomieniu i wykonaniu obliczeń
-q	Wyjście zostanie podane w uproszczonej formie, przykład: N 10 11 8 7 11 S 10 11 8 7 11 E 3 2 5 6 2 W 3 2 5 6 2
-t a	Obliczenia mają zostać wykonane dla wszystkich kolorów atutowych (oraz bez atu)
-d PBN	Informacja o tym, że rozdanie zostaje podane w formacie pbn
-c	Informacja o tym, że następnym argumentem będzie rozdanie w formacie podanym jako poprzedni parametr

Po wykonaniu obliczeń, wyniki trafiają do pliku *trickboard.txt*. Aby je z niego odczytać, wykorzystana zostaje funkcja *retrieve_table*, która wczytuje zawartość pliku o podanej nazwie, następnie ją parsuje, uzyskując listę list z elementami typu string, a następnie przekształca tak uzyskaną listę na słownik: *{ 'S' : [a1, b1, c1, d1, e1], 'W' : [a2, b2, c2, d2, e2], 'E' : [a3, b3, c3, d3, e3], 'N' : [a4, b4, c4, d4, e4] }*. Zmienne z przedrostkami: „a", „b", „c", „d" oraz „e" zawierają informacje o liczbie lew, które z ręki gracza można wziąć przy grze w poszczególne kolory (kolejno w trefle, kara, kiery, piki i bez atu) według programu BC.

44 Zmienna ta jest sformatowana według formatu pbn.

Słownik zwracany przez *retrieve_table* zapisywany jest jako zmienna *self.tricks* w obiekcie klasy *Deal* i dalsze obliczenia są już wykonywane w oparciu o słowniki uzyskane dla każdego z rozegranych rozdań.

4.1.3. Opis klas *Experiment* i *Deal*

4.1.3.1. Klasa *Experiment*

Obiekt klasy *Experiment* jest powoływany w celu przeprowadzenia dowolnego eksperymentu. Z klasy tej dziedziczy osiem podklas, z których każda odpowiada innemu typowi generowanych rozdań – zostały one opisane w podrozdziale 4.1.1. Wszystkie operacje na rozdaniach wykonywane są za pośrednictwem obiektów klasy *Experiment*. Podklasy różnią się jedynie trzema metodami: *generate*, *set_parameters*, oraz *prepare_report*, a różnice te wynikają z tego, że poszczególne typy eksperymentów wymagają użycia innych zestawów parametrów. Atrybuty klasy *Experiment* zostały opisane w tabeli 4.5, natomiast wszystkie metody tej klasy – w tabeli 4.6.

W obiekcie klasy *Experiment* przechowywane są również wszystkie wygenerowane na jego potrzeby rozdania (układy kart oraz rozkłady lew uzyskane przez uruchomienie funkcji *beling_engine*).

Table 4.5

Nazwa atrybutu	Opis
self.number	Unikalny numer identyfikacyjny dla danego eksperymentu.
self.boards	Lista rozdań wygenerowanych w ramach eksperymentu.
self.description	Opis / tytuł eksperymentu.
self.results	Aktualne rezultaty eksperymentu (średni rozkład lew w eksperymencie).
self.tries	Liczba prób, które mają zostać podjęte przy generowaniu rozdania (lub ręki dla każdego z graczy), zanim algorytm przerwie działanie i zwróci informację o niepowodzeniu.

Tabela 4.6

Nazwa metody	Opis
__init__	Konstruktor klasy Experiment.
calculate_expected_tricks	Odczytuje wyniki dla wszystkich rozegranych rozdań zgodnych z ustawionymi parametrami, a następnie oblicza średni rozkład lew

Nazwa metody	Opis
	w eksperymencie i zapisuje go do atrybutu self.results.
generate_deals(N)	Generuje N rozdań według ustawionych parametrów i dodaje je do listy rozdań przechowywanej w atrybucie self.boards.
generate_and_run(N)	Działa tak samo jak generate_deals, ale po wygenerowaniu rozdania zostają od razu rozegrane.
run(N)	Rozgrywa N nierozegranych rozdań z self.boards. Jeśli takich rozdań będzie mniej niż N, to zostaną wygenerowane dodatkowe rozdania aby uzupełnić do tej liczby.
run_all	Rozgrywa wszystkie nierozegrane rozdania z self.boards.
get_correct_deals	Dla każdego rozdania z self.boards sprawdza, czy jest ono poprawne (czyli spełnia ustawione parametry), a następnie zwraca liczbę takich rozdań[45].
get_played_deals	Zwraca liczbę rozegranych rozdań.
get_distribution	Na wejściu podawany jest kolor (C) oraz gracz rozgrywający (P). Metoda tworzy wektor (D) o długości 14, w którym zapisuje rozkład lew, które gracz P ma do wzięcia w kolor C, w oparciu o wszystkie rozegrane rozdania.
get_average_points	Oblicza średnią liczbę punktów dla każdego z graczy na podstawie poprawnych rozdań, następnie zwraca wartości średnie jako wektor.
get_average_cards	Oblicza średnią liczbę kart w każdym kolorze dla każdego gracza, następnie wyświetla obliczone wartości.
get_dist_distribution	Dla każdego z graczy tworzy słownik, a następnie zapisuje w nim wszystkie pojawiające się układy wraz z liczbą ich wystąpień i je wyświetla.
show_tricks	Dla każdego z rozdań wyświetla rozkład lew.
get_results	Wywołuje calculate_expected_tricks, następnie zwraca słownik przechowywany w atrybucie self.results, z wartościami zaokrąglonymi do dwóch miejsc po przecinku.
get_number	Zwraca numer identyfikacyjny eksperymentu przechowywany w self.number (numer ten jest nadawany jednorazowo dla każdego eksperymentu i jest unikalny).
save_report	Generuje raport z eksperymentu za pomocą metody prepare_report, a następnie zapisuje go do pliku o nazwie „experiment_[numer].txt” w katalogu Experiments.
save_results	Zapisuje liczbę poprawnych rozdań oraz słownik przechowywany w atrybucie self.results, do pliku o nazwie „results_[numer].txt” w katalogu Experiments.
save_experiment	Za pomocą funkcji dump z pakietu pickle zapisuje cały

45 Ta metoda jest wykorzystywana do obliczania średniego rozkładu lew w eksperymencie, w metodzie calculate_expected_tricks. Ponieważ generatory czasami mogą zwrócić rozdanie, które nie spełnia ustalonych parametrów, takie rozdania są pomijane w obliczeniach.

Nazwa metody	Opis
	eksperyment jako „exp_[numer].obj” w katalogu Experiments.
generate	Generuje rozdanie według ustalonych parametrów i dodaje je do listy przechowywanej w atrybucie self.boards.
prepare_report	Na podstawie ustawionych parametrów i wyników rozegranych rozdań tworzy raport z eksperymentu.
set_parameters	Za pomocą tej metody ustawiane są parametry eksperymentu takie jak: liczba prób (self.tries), karty jednego (self.h) lub dwóch graczy (self.s i self.n), opis parametrów dla innych rąk (self.a, self.b, self.c, self.d) oraz informacje do ręki którego z graczy należy dany opis (np. self.whob).

4.1.3.2. Klasa Deal

Każdy obiekt klasy *Deal* stanowi reprezentację jednego rozdania brydżowego. W każdym obiekcie klasy *Experimen*t, w atrybucie self.boards znajduje się lista obiektów typu *Deal,* czyli rozdań wygenerowanych według ustawionych parametrów. Atrybuty, które może posiadać każdy obiekt klasy *Deal*, przedstawione są w tabeli 4.7, natomiast metody klasy *Deal* – w tabeli 4.8.

Tabela 4.7

Nazwa atrybutu	Opis
self.board	W tym atrybucie przechowywane jest rozdanie w formie listy czterech wektorów o długości 13, które kolejno odpowiadają rękom graczy S, W, N i E.
self.correct	Zawiera informację o tym, czy rozdanie spełnia ustalone w eksperymencie parametry.
self.is_played	Zawiera informację o tym, czy dane rozdanie zostało już rozegrane przez program BridgeCalculator.
self.tricks	Jeśli rozdanie zostało rozegrane, w atrybucie self.tricks przechowywany będzie słownik z informacją o rozkładzie lew.

Tabela 4.8

Nazwa metody	Wejście	Wyjście	Opis
__init__	[R], C	brak	Układ kart podany na wejściu (R) zostaje przypisany do self.board, a informacja o tym, że rozdanie spełnia warunki postawione w eksperymencie (C) do self.correct. Atrybut self.is_played zostaje ustawiony na False.
get_tricks	C, P	brak	Sprawdza, czy self.is_played nie jest równe False, jeśli tak to uruchamia metodę run. Jeśli nie, to zwraca liczbę lew, którą w danym rozdaniu do wzięcia ma gracz P, w kolorze C.
run	brak	brak	Sprawdza czy self.correct jest równe True, a self.is_played jest równe False. Jeśli tak, to uruchamia funkcję beling_engine dla danego rozdania, następnie odczytuje wyniki za pomocą funkcji retrieve_table i zapisuje ją w self.tricks, oraz ustawia self.is_played na True.
print_deal	brak	brak	Wyświetla ręce wszystkich graczy.

4.1.4. Funkcje pomocnicze

Poza metodami klas *Experiment* i *Deal*, w programie pojawiają się także funkcje pomocnicze, opisane w tabeli 4.9.

Tabela 4.9

Nazwa funkcji	Opis
load_experiment	Funkcja służy do wczytywania wcześniej zapisanego eksperymentu.
get_score	Na wejściu zostaje podana wysokość kontraktu, liczba wziętych lew, liczba lew wymaganych do realizacji kontraktu, kolor atutowy, informacja o tym czy strona rozgrywająca była przed partią czy po, oraz czy kontrakt był skontrowany albo zrekontrowany. Zwracany jest zapis uzyskany przez stronę rozgrywającą.
get_scores_vector	Na wejściu zostaje podana wysokość kontraktu, kolor atutowy, informacja o tym czy strona rozgrywająca była przed partią czy po, oraz czy kontrakt był skontrowany albo zrekontrowany. Zwracany jest wektor wyników, który zawiera zapisy uzyskiwane przez stronę rozgrywającą dla każdej liczby wziętych lew (od 0 do 13).
get_frequency	Na wejściu podawane są parametry ręki oraz liczba rozdań do wygenerowania. Po wygenerowaniu rozdań podawana jest informacja o tym jak często jedna z wygenerowanych w tych rozdaniach rąk spełniała te parametry.
get_frequency_for_each	Na wejściu podawany jest zestaw parametrów ręki w formie listy oraz liczba rozdań do wygenerowania. Po wygenerowaniu rozdań podawana jest informacja o tym jak często występowały warianty ręki spełniające poszczególne zestawy parametrów.

Nazwa funkcji	Opis
loc_min	Na wejściu podawane są parametry ręki, a następnie zwracane są minimalne wymagania.
mass_loc_min	Na wejściu podawana jest lista parametrów dla ustalonej ręki (różne warianty), następnie dla każdego wariantu jest wywoływana funkcja loc_min.
normalize_rules_type_X	Na wejściu podawane są np. ręka jednego z graczy oraz zestawy parametrów opisujące ręce pozostałych graczy. Normalizacja polega na tym, że maksimum każdego parametru dla każdej ręki zostaje ograniczone przez sumę minimów tych samych parametrów ze wszystkich pozostałych rąk. Np. gdy w podanej ręce gracza S jest 15 punktów, a gracz W może mieć 18+ punktów (domyślnie: 18 – 37), w tym przypadku maksimum punktów dla gracza W zostanie ustawione na 40 -15 (punkty S), czyli 25pkt.[46]

4.1.5. Automatyczna weryfikacja i normalizacja

4.1.5.1. Weryfikacja ograniczeń

W przypadku gdy użytkownik wprowadzi zestaw parametrów, na podstawie których wygenerowanie rozdania jest niemożliwe, program wychwyci to na etapie weryfikacji i eksperyment nie zostanie uruchomiony. Procedura ta jest przeprowadzana w sposób automatyczny przed rozpoczęciem generowania rozdań.

Przykład: *Dane są: ręka S: ♠W543 ♥W105 ♦K ♣KW1082, oraz zestawy ograniczeń dla rąk: W (6-10 punktów, 6-7 kierów, 0-4 w pozostałych kolorach), E (0-12 punktów, 0-5 pików, 0-4 kiery, 0-6 kar i 0-6 trefli) oraz N (12-15 punktów, co najmniej 4 piki, 0-3 kiery, 0-6 kar i 0-6 trefli lub co najmniej 16 punktów i układ dowolny). Aby dokonać weryfikacji podanych ograniczeń, należy wykonać następujące kroki:*

1. $B=[40,13,13,13,13]$
2. $M_s=[9,4,3,1,5], M_N=[16,4,0,0,0], M_W=[6,0,6,0,0], M_E=[0,0,0,0,0]$
3. $M_{SUMA}=M_s+M_N+M_W+M_E=[31,8,9,1,5]$
4. $T_M=B-M_{SUMA}=[9,5,4,12,8]$

Wektor B to zasoby dostępne na początku (40 punktów oraz po 13 kart w każdym kolorze). Wektory M_X reprezentują największe dolne ograniczenia przedziałów (po wszystkich wariantach dla danego zestawu ograniczeń). Wektor M_{SUMA} reprezentuje wszystkie zarezerwowane zasoby, natomiast zasoby, które pozostały wolne reprezentuje wektor T_M . Ponieważ wszystkie wartości w wektorze T_M są większe (lub równe) od

46 Normalizacja została bardziej szczegółowo opisana w podrozdziale 4.1.5.

zera, pierwszy etap weryfikacji zakończył się sukcesem. Kolejne trzy kroki przedstawiają weryfikację ograniczeń górnych:

$$5.\ U_S=[9,4,3,1,5], U_N=[15,13,13,13,13], U_W=[10,4,7,4,4], U_E=[12,5,4,6,6]$$

$$6.\ U_{SUMA}=U_S+U_N+U_W+U_E=[46,26,26,24,28]$$

$$7.\ T_U=U_{SUMA}-B=[6,13,13,11,15]$$

Wektory U_X reprezentują najmniejsze górne ograniczenia przedziałów (po wszystkich wariantach dla danego zestawu ograniczeń). Wektor U_{SUMA} to suma tych ograniczeń górnych, natomiast wektor T_U reprezentuje nadwyżkę nad zasobami bazowymi. Ponieważ wszystkie wartości w wektorze T_U są większe (lub równe) od zera, drugi etap weryfikacji również zakończył się sukcesem.

W przypadku gdy ograniczenia dla którejś z rąk składają się z wzajemnie wykluczających się wariantów, weryfikacja również może zakończyć się niepowodzeniem. Przykładowo, gdy ręka S ma 5 pików i 5 trefli lub 6 kierów, zarezerwowanych zostanie aż 16 kart. Jeśli ograniczenia dla pozostałych rąk będą bardzo precyzyjne, a zarazem nie będą wykluczały żadnego wariantu ręki S, może dojść do sytuacji, w której co najmniej jedna wartość w wektorze T_M będzie ujemna. Najprostszym rozwiązaniem tego problemu jest podzielenie eksperymentu na etapy (w tym przypadku na dwa). W każdym z nich ograniczenia ręki S zostałyby sprowadzone do jednego wariantu. Należy wyznaczyć rozkład prawdopodobieństwa dla każdego z tych wariantów (można to zrobić poprzez wygenerowanie rozdań przy założeniu, że ręka S nie jest znana, wybrać z nich wszystkie rozdania, w których S miał jeden z wymienionych wyżej wariantów, a następnie sprawdzić ile razy wystąpił każdy wariant. Jeśli np. wygenerowano 1000 rozdań, z czego 66 razy S miał 6 kierów, a 34 razy piki i trefle w układzie 5-5, to na podstawie tego można przyjąć rozkład prawdopodobieństwa: 0,66 i 0,34). Zakładając, że cały eksperyment będzie składał się z 1000 rozdań, najpierw trzeba ustawić ograniczenia z pierwszym wariantem ręki S i wygenerować 340 rozdań, następnie zmienić wariant ręki S i wygenerować kolejne 660 rozdań. Jeśli więcej niż jedna ręka jest zdefiniowana w podobny sposób, oczywiście dojdzie do dalszej komplikacji obliczeń.

4.1.5.2. Normalizacja ograniczeń

Ponieważ parametry podawane są w formie przedziałów (punktów, kart w danym kolorze), aby użytkownik nie musiał ręcznie przeprowadzać dodatkowych obliczeń, w projekcie zastosowana została również normalizacja zestawu ograniczeń (jest to funkcja pomocnicza opisana w podrozdziale 4.1.4), dzięki której górna granica każdego przedziału

jest poprawiana na taką, która nie spowoduje, że znalezienie rozdania spełniającego zadane ograniczenia będzie niemożliwe.

Jeśli ograniczenia dla generowanych rozdań zostały zweryfikowane pozytywnie, kolejny krok polega na poprawieniu wartości maksymalnych. Załóżmy na przykład, że w ręce S musi znajdować się co najmniej 5 kierów. Oznacza to, że pozostałe ręce mogą mieć razem co najwyżej 8 kierów. W przypadku gdy jakaś ręka jest znana, to pierwszy etap procedury jest taki sam jak przy weryfikacji. Dla każdej ręki sprawdzane jest, ile punktów i kart w poszczególnych kolorach należy zarezerwować dla pozostałych rąk, a następnie ta liczba jest odejmowana od 40 (w przypadku punktów), lub od 13 (w przypadku kart w kolorze). Jeśli okaże się, że maksymalna liczba punktów (kart w kolorze) zapisana w ograniczeniach dla danej ręki jest większa, niż uzyskana liczba, wtedy zostaje zastąpiona tą liczbą.

Przykład: *S ma dokładnie 5 kierów. N ma od 5 do 8 kierów, natomiast E ma co najmniej 2 kiery. Aby ustalić górną granicę przedziału (górną granicę liczby kierów, oznaczamy jako* $U_{\heartsuit}$*, dolną jako* $M_{\heartsuit}$ *), dla każdego z graczy wykonujemy następujące obliczenia:*

$$1.\ U_{\heartsuit}(N)=13-(M_{\heartsuit}(S)+M_{\heartsuit}(E)+M_{\heartsuit}(W))=13-(5+2+0)=6$$

$$2.\ U_{\heartsuit}(E)=13-(M_{\heartsuit}(S)+M_{\heartsuit}(N)+M_{\heartsuit}(W))=13-(5+5+0)=3$$

$$3.\ U_{\heartsuit}(W)=13-(M_{\heartsuit}(S)+M_{\heartsuit}(N)+M_{\heartsuit}(E))=13-(5+5+2)=1$$

Oznacza, że N może mieć od 5 do 6 kierów, E może mieć od 2 do 3 kierów, natomiast W może mieć co najwyżej jednego kiera.

Takie działania przeprowadzane są dla punktów i dla każdego koloru, a ich celem jest poprawa wydajności generatora rozdań (który dzięki normalizacji unika „ślepych zaułków” polegających na tym, że znajdzie rękę spełniającą ograniczenia dla jednego z graczy, ale jednocześnie wykluczającą możliwość znalezienia ręki pasującej do ograniczeń innego z graczy).

4.1.6. Przykład zastosowania

W tym rozdziale zostało opisane przykładowe zastosowanie narzędzi przedstawionych w podrozdziałach 4.1.1 – 4.1.4. Rozwiązywany problem zostanie omówiony w podrozdziale 4.2.3.3, jako eksperyment „8”.

Eksperyment „8” (ustawianie parametrów i wywołanie)

```
1   desc = „problem 6 z 4”
2   h = [11, 14, 15, 19, 24, 28, 32, 37, 40, 41, 42, 44, 48]
4   n = [[[5, 10], [6, 7], [0, 13], [0, 6], [0, 13]]]
5   w = [[[0, 9], [0, 5], [0, 5], [0, 3], [0, 6]]]
6   e = [[[12, 17], [0, 4], [0, 4], [5, 13], [0, 13]],
7        [[12, 17], [0, 4], [0, 4], [4, 13], [5, 13]],
8        [[12, 17], [0, 1], [4, 4], [4, 4], [4, 4]],
9        [[12, 17], [4, 4], [0, 1], [4, 4], [4, 4]],
10       [[12, 17], [4, 4], [4, 4], [4, 4], [0, 1]]]
11  deals = 1000
12  exp = Type_Three(desc)
13  exp.set_parameters(h, n, w, e, 500)
14  exp.generate_and_run(deals)
15  exp.save_experiment()
16  a = exp.get_distribution("N", "S")
17  b2 = get_scores_vector(4, "S", True, False, False)
18  b3 = get_scores_vector(3, "S", True, False, False)
19  print dot_product(b, b2), " 4S z reki N"
20  print dot_product(b, b3), " 3S z reki N"
```

W liniach od 1 do 11 zdefiniowane zostały parametry rozdania (opis, następnie karty gracza S, potem od linii 3 do 10 parametry rąk pozostałych graczy – ręka gracza E ma pięć możliwych wariantów). W linii 11 ustalona została liczba rozdań. Od 12 do 15 linii: utworzony zostaje obiekt klasy *Type_Three*, ustawione zostają parametry (500 to liczba prób, zapisywana w atrybucie *self.tries*), następnie zostaje wygenerowanych i rozegranych 1000 rozdań. W linii 15 eksperyment zostaje zapisany.

W linii 16 z eksperymentu zostaje odzyskany rozkład lew w piki z ręki N. Następnie dwukrotnie wywołana jest funkcja *get_scores_vector* dla kontraktów 4♠ i 3♠. Ostatnie dwie linie wyświetlają wartości oczekiwanego zapisu dla każdego z kontraktów.

4.2. Badania statystyczne – przykłady

W rozdziale pierwszym zaznaczono już, że żaden eksperyment nie wystarczy, aby stwierdzić, że dana hipoteza została w pełni zweryfikowana, ponieważ w tym celu należałoby sprawdzić wszystkie rozdania spełniające określone warunki, których na ogół będzie bardzo wiele[47]. Rezultaty eksperymentu mogą jedynie wskazywać, że badania statystyczne sugerują poprawność hipotezy, a w przypadku gdy rezultaty nie są zgodne z oczekiwanymi, jej brak. Często w celu weryfikacji hipotezy konieczne będzie przeprowadzenie więcej niż jednego eksperymentu.

Pierwszy eksperyment oznaczony jako „0” będzie dotyczył weryfikacji wiarygodności generatora rozdań *Type_Zero*. Kolejne eksperymenty zostały podzielone na dwie klasy: pierwszą z nich będą stanowiły próby weryfikacji „praw brydżowych”, natomiast drugą – próby rozwiązywania problemów decyzyjnych w licytacji.

4.2.1. Eksperyment “0”

Hipoteza: *Generator rozdań Type_Zero jest wiarygodny.*

Cel: Celem tego eksperymentu jest zbadanie, czy rozdania generowane przez generator *Type_Zero* mają rozkład zgodny z rozkładami matematycznymi (dla prób o wielkości: 1000, 10000 i 100000 rozdań).

Opis: W ramach eksperymentu zostaną wygenerowane trzy zbiory rozdań (o wielkości 1000, 10000, 100000), a następnie zostaną zbadane ich własności: średnia siła każdej z czterech rąk, średnia liczba kart w każdym kolorze dla każdej z czterech rąk, rozkład siły rąk oraz rozkład układów kart. Uzyskane rezultaty zostaną porównane ze znanymi rozkładami matematycznymi[48].

Wejścia: Liczby rozdań, które mają zostać zbadane (wielkość próby).

Wyjście: Tabele: 4.10, 4.11, 4.12 i 4.13

47 Liczba wszystkich możliwych rozdań brydżowych: 53 644 737 765 488 792 839 237 440 000. Liczba wszystkich rąk, jakie gracz może otrzymać: 635 013 559 600. Liczba rąk, które może otrzymać gracz, gdy ustalone są karty jednego dowolnego gracza: 8 122 425 444. Liczba rąk, które mogą otrzymać trzeci i czwarty gracz, gdy ustalone są karty dwóch pierwszych graczy: 10 400 600.

48 Do porównania wzięte zostały rozkłady z: http://www.bridgehands.com/P/Probability_Hand_Distribution.htm oraz http://www.bridgehands.com/P/Probability_HCP.htm .

Tabela 4.10

Gracz / Rozdania	S	W	N	E	σ
1000	9.99	9.998	9.937	10.075	0.0984784
10000	9.9708	9.9358	9.9951	10.0983	0.121083
100000	10.01338	10.00068	9.99811	9.98783	0.0181980
Średnia oczekiwana	10	10	10	10	0

Tabela 4.11

Gracz / rozdania	S	W	N	E	σ
1000 (♠)	3.273	3.252	3.301	3.174	0.0943928
1000 (♥)	3.218	3.245	3.265	3.272	0.0419285
1000 (♦)	3.196	3.234	3.261	3.309	0.0823043
1000 (♣)	3.313	3.269	3.173	3.245	0.1014100
10000 (♠)	3.2495	3.2335	3.2582	3.2588	0.0204249
10000 (♥)	3.2612	3.262	3.2318	3.245	0.0250135
10000 (♦)	3.2684	3.2637	3.2369	3.231	0.0325401
10000 (♣)	3.2209	3.2408	3.2731	3.2652	0.0411837
100000 (♠)	3.24448	3.24853	3.25793	3.24906	0.0098189
100000 (♥)	3.24906	3.25448	3.25021	3.24625	0.0059211
100000 (♦)	3.24972	3.25674	3.24596	3.24758	0.0082270
100000 (♣)	3.25674	3.24025	3.2459	3.25711	0.0144170
Średnia oczekiwana	3.25	3.25	3.25	3.25	0

Tabela 4.12

Punkty	Frekwencja oczekiwana[49]	Frekwencja: (1000 rozdań)	Frekwencja: (10000 rozdań)	Frekwencja: (100000 rozdań)
0	0.0036	0.0035	0.004025	0.0037275
1	0.0079	0.00775	0.00765	0.00786
2	0.0136	0.011	0.013975	0.0135
3	0.0246	0.02425	0.025075	0.02442
4	0.0385	0.041	0.038225	0.038305
5	0.0519	0.05375	0.051725	0.05201
6	0.0655	0.0665	0.0658	0.065625
7	0.0803	0.081	0.0798	0.080195

49 Na podstawie: http://www.bridgehands.com/P/Probability_HCP.htm.

Punkty	Frekwencja oczekiwana	Frekwencja: (1000 rozdań)	Frekwencja: (10000 rozdań)	Frekwencja: (100000 rozdań)
8	0.0889	0.08675	0.087625	0.0894075
9	0.0936	0.086	0.0946	0.0938175
10	0.0941	0.0985	0.095675	0.093895
11	0.0894	0.0915	0.0875	0.0889775
12	0.0803	0.07875	0.079575	0.08047
13	0.0691	0.06925	0.0691	0.06884
14	0.0569	0.0075	0.056525	0.056925
15	0.0442	0.00325	0.044725	0.0445
16	0.0331	0.061	0.03235	0.0326625
17	0.0236	0.04625	0.0247	0.02402
18	0.0161	0.02825	0.016625	0.01635
19	0.0104	0.0265	0.010425	0.010205
20	0.0064	0.01575	0.006525	0.0063575
21	0.0038	0.008	0.003475	0.0036725
22	0.0021	0.00275	0.002025	0.0021225
23	0.0011	0.00075	0.001325	0.0010725
24	0.0006	0.00025	0.0005	0.00056
25	0.0003	0.00025	0.0002	0.0002675
σ	0.00	0.0776396483763	0.0036	0.0011575242978

Tabela 4.13

Układ[50]	Frekwencja oczekiwana[51]	Frekwencja: (1000 rozdań)	Frekwencja: (10000 rozdań)	Frekwencja: (100000 rozdań)
4432	21.6	0.214	0.216575	0.2133825
5332	15.5	0.1515	0.159775	0.156695
5431	12.9	0.1365	0.12945	0.1296875
5422	10.6	0.107	0.10395	0.10579
4333	10.5	0.1105	0.106775	0.1052325
6322	5.6	0.05875	0.05575	0.056155
6421	4.7	0.043	0.0464	0.0469025
6331	3.5	0.03225	0.0358	0.03512
5521	3.2	0.03	0.02995	0.0316625
4441	3.0	0.028	0.0295	0.029735

50 Do porównania zostały wybrane układy o największej frekwencji.
51 Na podstawie: http://www.bridgehands.com/P/Probability_Hand_Distribution.htm.

Układ	Frekwencja oczekiwana	Frekwencja: (1000 rozdań)	Frekwencja: (10000 rozdań)	Frekwencja: (100000 rozdań)
7321	1.9	0.01925	0.0184	0.01872
6430	1.3	0.015	0.013075	0.0133
5440	1.2	0.013	0.0125	0.0126025
5530	0.90	0.00925	0.008425	0.0090325
6511	0.71	0.0055	0.00635	0.007055
σ	0	0.0122702893201	0.0061408977357	0.0033259124688

Wnioski: Wyniki zaprezentowane w tabelach od 4.10 do 4.13 pokazują, że wraz ze zwiększaniem próby, odchylenie od wartości oczekiwanych maleje, co oznacza, że generowane rozdania mają własności rozdań losowych. Zatem hipoteza o wiarygodności generatora Type_Zero została zweryfikowana pozytywnie.

4.2.2. Weryfikacja praw brydżowych

4.2.2.1 Eksperyment “1”

Hipoteza: *Jeżeli jeden z graczy ma wyraźnie silniejszą rękę od partnera, to bardziej korzystne jest rozgrywanie kontraktu z jego ręki.*

Cel: Powyższa hipoteza to jedno z najbardziej znanych „praw brydżowych”, oparte na intuicji – jeżeli silniejsza ręka nie jest znana obrońcom, to para będąca na wiście jest w trudniejszej sytuacji, gdyż posiada mniej istotnych informacji. Ponieważ program stosuje algorytm do rozgrywki w otwarte karty, wyniki pozwolą ustalić, czy jest to ogólna cecha rozdań danego typu.

Opis: W ramach tego eksperymentu zostanie wygenerowanych 4000 rozdań, w których gracz S ma rękę w sile 15 do 17 punktów, natomiast gracz N – od 7 do 9. Następnie zostanie wyznaczony rozkład lew w eksperymencie, na podstawie którego obliczona zostanie średnia liczba lew do wzięcia na linii NS we wszystkie kolory i bez atu. Jeżeli dla każdego koloru (i bez atu) średnia liczba lew będzie wyższa przy rozgrywce z ręki S, niż z ręki N, to hipoteza zostanie zweryfikowana pozytywnie.

Wejście: Liczba rozdań (4000), parametry dla ręki S (15-17 punktów, układ dowolny), parametry dla ręki N (7-9 punktów, układ dowolny).

Wyjście: Tabela 4.14

Tabela 4.14

Rozgrywający	♣	♦	♣	♠	BA
S	7.812	7.799	7.813	7.854	7.982
N	7.776	7.768	7.785	7.825	7.936

Wnioski: Dla każdego koloru rozgrywka z ręki S średnio przynosi większą liczbę lew dla pary NS, niż rozgrywka z ręki N. Różnica ta wynosi około 0,03 lewy na rozdanie w przypadku gry w kolor, oraz prawie 0,05 lewy na rozdanie przy grze w bez atu. W ten sposób stwierdziliśmy, że średnia liczba lew do wzięcia z ręki silniejszej (S) jest większa od liczby lew do wzięcia z ręki słabszej (N), zatem uzyskane wyniki pozwalają na przyjęcie hipotezy.

4.2.2.2. Eksperyment “2”

Hipoteza: *Jeżeli para dysponuje dwoma równorzędnymi kolorami do gry, to najbardziej korzystna jest rozgrywka z ręki silniejszej (patrz: Eksperyment “1”), w kolor należący do ręki słabszej.*

Cel: W eksperymencie „1” potwierdzona została hipoteza, że bardziej korzystna jest rozgrywka z ręki silniejszej. Eksperyment „2” to próba rozszerzenia tej hipotezy o jeden dodatkowy warunek.

Opis: W ramach tego eksperymentu zostanie wygenerowanych 4000 rozdań, w których gracz S ma rękę w sile od 15 do 17 punktów, dokładnie pięć pików i trzy kiery, natomiast gracz N od 7 do 9 punktów, dokładnie trzy piki i pięć kierów. Następnie zostanie wyznaczony rozkład lew w eksperymencie, na podstawie którego obliczona zostanie średnia liczba lew na kontrakty pikowe i kierowe grane na linii NS. Hipoteza zostanie zweryfikowana pozytywnie, jeżeli największa średnia liczba lew będzie do wzięcia w kolorze kierowym (kolor słabszej ręki, czyli N), na kontrakt rozgrywany z ręki S.

Wejście: Liczba rozdań (4000), parametry dla ręki S (15-17 punktów, 5 pików i 3 kiery), parametry dla ręki N (7-9 punktów, 3 piki i 5 kierów).

Wyjście: Tabela 4.15

Tabela 4.15

Rozgrywający	♣	♦	♥	♠	BA
S			9.665	9.637	
N			9.637	9.616	

Wnioski: Z tabeli 4.15 wynika, że średnio najwięcej lew jest do wzięcia z ręki S (silniejszej), na kontrakt kierowy, natomiast najmniej na kontrakt pikowy rozgrywany z ręki N. Oznacza to, że hipoteza została zweryfikowana pozytywnie.

4.2.2.3. Eksperyment "3"

Hipoteza: *Rozgrywka w kolor rozłożony 4-4 jest bardziej opłacalna, niż w kolor 5-3, jeśli para ma takie kolory.*

Cel: To „prawo brydżowe" na stałe weszło już do kanonów brydżowej wiedzy (na jego podstawie budowane są niektóre konwencje licytacyjne). Hipoteza dotyczy sytuacji, w której para z licytacji wie o tym, że dysponuje dwoma nadającymi się do gry kolorami, ale jeden z nich jest rozłożony 5-3, a drugi 4-4. Chodzi o sprawdzenie, czy faktycznie granie w kolor rozłożony 4-4 pozwala na wzięcie (średnio) większej liczby lew.

Opis: W ramach tego eksperymentu zostanie wygenerowanych 4000 rozdań, w których gracz S ma rękę w dowolnej sile, dokładnie pięć pików i dokładnie cztery kiery, natomiast gracz N ma rękę w dowolnej sile, dokładnie trzy piki i dokładnie cztery kiery. Następnie zostanie wyznaczony rozkład lew w eksperymencie, z którego odczytane zostaną średnie liczby lew do wzięcia w kiery i piki, z rąk S oraz N. Jeżeli średnia liczba lew do wzięcia w kolorze kierowym (4-4) będzie większa od średniej liczby lew w kolorze pikowym (5-3), niezależnie od ręki z której kontrakt był rozgrywany, to hipoteza zostanie zweryfikowana pozytywnie.

Wejście: Liczba rozdań, parametry dla ręki S (dowolna siła, 5 pików i 4 kiery), parametry dla ręki N (dowolna siła, 3 piki, 4 kiery).

Wyjście: Tabela 4.16

Tabela 4.16

Rozgrywający	♣	♦	♥	♠	BA
S			8.22	8.144	
N			8.219	8.148	

Wnioski: Z tabeli 4.16 wynika, że na próbie 4000 rozdań przewaga gry w kiery nad grą w piki, i to niezależnie od wyboru ręki z której rozgrywany był kontrakt, została potwierdzona. W przypadku rozgrywki z ręki S, różnica wynosi 0,076 lewy na rozdanie, natomiast w przypadku rozgrywki z ręki N – 0,071. Zatem hipoteza o tym, że granie w kolor rozłożony 4-4 jest bardziej korzystne (biorąc pod uwagę liczbę lew do wzięcia), niż w kolor 5-3, została zweryfikowana pozytywnie.

4.2.2.4. Eksperyment „4”

Hipoteza: *Gdy rozdania są losowe, średnia liczba lew do wzięcia z ręki każdego gracza przy rozgrywce w każdy kolor oraz bez atu, dąży do pewnej liczby (różnej dla rozgrywki w kolor i dla rozgrywki w bez atu).*

Cel: Założenie jest następujące – skoro rozdania są losowe, a poszczególne wartości np. średnia liczba punktów dla każdej ręki, czy też średnia liczba kart w każdym kolorze dla każdej ręki są zbieżne do konkretnych wartości, to również średnie liczby lew do wzięcia w kolor i bez atu mogą (a może nawet: powinny) zbiegać do pewnych wartości. Za pomocą tego eksperymentu podjęta zostanie próba wyznaczenia takich wartości. Jeśli takie wartości istnieją, to mogą być one również wykorzystane do badania wiarygodności generatorów rozdań.

Opis: W ramach eksperymentu zostanie wygenerowanych (za pomocą generatora Type_Zero, ponieważ w podrozdziale 4.2.1 została potwierdzona hipoteza o jego wiarygodności) i rozegranych 1000 rozdań, następnie zostanie wyznaczony rozkład lew w eksperymencie. Eksperyment ten zostanie powtórzony dla próby 10000 rozdań.

Wejście: Liczba rozdań (1000).

Wyjście: Tabela 4.17

Tabela 4.17

Rozgrywający	♣	♦	♥	♠	BA
S	6.156	6.382	6.313	6.366	6.071
E	6.439	6.232	6.307	6.227	5.961
W	6.43	6.221	6.296	6.217	5.955
N	6.169	6.375	6.307	6.373	6.07
Średnia	6.2985	6.3025	6.30575	6.29575	6.01425

Średnia liczba lew do wzięcia w dowolny kolor inny niż BA: 6.300625

Wnioski: Wyniki przedstawione w tabeli 4.17 pokazują, że dla próby 1000 rozdań średnia liczba lew do wzięcia w kolor oscyluje wokół 6,3, natomiast średnia liczba lew do wzięcia w bez atu jest bliska 6,0.

Wejście: Liczba rozdań (10000).

Wyjście: Tabela 4.18

Tabela 4.18

Rozgrywający	♣	♦	♥	♠	BA
S	6.304	6.328	6.317	6.312	6.037
E	6.29	6.264	6.268	6.292	5.982
W	6.287	6.266	6.264	6.292	5.976
N	6.308	6.329	6.315	6.307	6.04
Średnia	6.29725	6.29675	6.291	6.30075	6.00875

Średnia liczba lew do wzięcia w dowolny kolor inny niż BA: 6.2964375

Wnioski: Wyniki przedstawione w tabeli 4.18 pokazują, że dla próby 10000 rozdań średnia liczba lew do wzięcia w kolor oscyluje wokół 6,3, natomiast średnia liczba lew do wzięcia w bez atu oscyluje wokół 6,0. Ponieważ hipoteza o losowości generowanych rozdań (eksperyment „0”) została potwierdzona, należy uznać, że średnie liczby lew do wzięcia w kolor i bez atu dążą do pewnych wartości. Na podstawie wyników przedstawionych w tabelach 4.17 oraz 4.18 można wyznaczyć te wartości: około 6,3 dla gry w kolor oraz około 6,0 dla gry w bez atu.

4.2.2.5 Eksperyment „5”

Hipoteza: *Rozgrywka w kolor rozłożony 4-4 jest bardziej opłacalna, niż w kolor 5-3, nawet gdy kontrakt rozgrywany jest ze słabszej ręki.*

Cel: W eksperymencie „1” pozytywnie została zweryfikowana hipoteza, że bardziej korzystna jest sytuacja, w której kontrakt rozgrywany jest z ręki silniejszej. Następnie, w eksperymencie „3” potwierdzono hipotezę o tym, że w przypadku gdy para posiada dwa równorzędne kolory, na kontrakt rozgrywany w kolor rozłożony 4-4 można wziąć średnio więcej lew, niż w kolor rozłożony 5-3. Postawiona hipoteza ma na celu sprawdzenie, które z tych dwóch praw jest „silniejsze”.

Opis: Eksperyment ten zostanie podzielony na dwie części, z których każda będzie miała identyczny przebieg. Najpierw wygenerowanych zostanie 4000 rozdań spełniających określone (dla każdej części eksperymentu trochę inne) parametry, następnie rozdania te zostaną rozegrane i wyznaczony zostanie rozkład lew. Z tego rozkładu zostaną odczytane średnie liczby lew do wzięcia na kontrakty kierowe i pikowe, z rąk graczy S i N.

Wejście (część I): Liczba rozdań (4000), siła i układ ręki S (15-17, 4♠ - 3♥), siła i układ ręki N (7-9, 4♠ - 5♥).

Wejście (część II): Liczba rozdań (4000), siła i układ ręki S (15-17, 4♠ - 5♥), siła i układ ręki N (7-9, 4♠ - 3♥).

Wyjście: Tabela 4.19.

Tabela 4.19

Rozgrywający	♣	♦	♥	♠	BA
S (I)			9.594	9.711	
N (I)			9.572	9.693	
S (II)			9.602	9.745	
N (II)			9.586	9.728	

Wnioski: Z tabeli 4.19 można odczytać, że niezależnie od tego, z której ręki rozgrywany jest kontrakt, średnia liczba lew do wzięcia w piki jest zawsze większa od

liczby lew do wzięcia w kiery. Najmniejsza średnia liczba lew do wzięcia w piki z ręki N wynosi 9,693 (gdy N ma układ 5-4), natomiast największa średnia liczba lew do wzięcia w kiery z ręki S wynosi 9,602 (gdy układ 5-4 posiada S). Oznacza to, że niezależnie od tego, który z graczy ma układ 5-4, a który 3-4 w tych kolorach, a także niezależnie od tego, która ręka jest silniejsza, rozgrywka w kolor rozłożony 4-4 jest średnio bardziej korzystna[52].

Wynika z tego, że „prawo silniejszej ręki" jest „słabsze" niż „prawo przewagi koloru 4-4 nad 5-3", zatem hipoteza została zweryfikowana pozytywnie.

4.2.3. Rozwiązywanie problemów decyzyjnych w licytacji

Eksperymenty opisane w podrozdziale 4.2.2 dotyczyły "praw brydżowych" o charakterze ogólnym, natomiast przedstawione w podrozdziale 4.2.3 dotyczyć będą problemów związanych z podejmowaniem decyzji w trakcie licytacji. Konkretnie, będą to przykłady wykorzystania programu do rozwiązywania wybranych problemów konkursowych z miesięcznika "Brydż" (lub podobnych), w których znana jest ręka jednego z graczy (S). W przypadku tych problemów zamiast hipotezy opis będzie dotyczył tego, na czym polega problem decyzyjny.

4.2.3.1. Eksperyment "6"

Problem: *Po otwarciu 1♦ (mogło być dane w sile 11-14 punktów), odpowiedzi 1♠ i rebidzie 2♠, z kartą typu: ♠K9754 ♥95 ♦KDW ♣D75 chcemy ustalić, czy bardziej korzystna jest dalsza licytacja czy spasowanie.*

Cel: Podana jest konkretna sytuacja licytacyjna, jedna ręka zdefiniowana przez otwarcie i jedna ręka znana (znane są jej wszystkie karty). Trzeba ustalić, jaka decyzja daje lepsze rezultaty. Pas będzie najbardziej korzystnym rozwiązaniem, jeżeli wartość oczekiwana wyniku za kontrakt 2♠ będzie większa niż za 4♠ (bo dalsza licytacja ma sens jedynie wtedy, gdy w perspektywie jest zagranie tego kontraktu).

52 Konsekwencją takiego wniosku jest to, że nawet po znalezieniu fitu 5-3 w kolorze starszym, warto również sprawdzić, czy w drugim kolorze starszym nie dysponuje się fitem 4-4 (analogicznie w kolorach młodszych). A także, że w sytuacji gdy jeden z graczy rozważa grę premiową, nawet po uzgodnieniu koloru starszego warto spróbować poszukać fitu 4-4 w kolorze młodszym. W żadnej z tych sytuacji nie należy się sugerować rozkładem siły na obu rękach.

Opis: W ramach tego eksperymentu zostanie wygenerowanych 4000 rozdań, w których gracz S ma rękę podaną w opisie problemu, natomiast N posiada rękę w sile 11-14 punktów, 4+ kara, 4 piki. Następnie zostanie wyznaczony rozkład lew do wzięcia w piki z ręki S. Dla uzyskanych wyników zostaną obliczone i porównane wartości oczekiwane rezultatów, które zostałyby osiągnięte przy rozgrywce kontraktów 2♠ i 4♠.

Wejście: Liczba rozdań, ręka S, parametry dla ręki N (11-14 punktów, 4 piki i 5 kar lub 4 piki i 4 kara w układzie 4441[53]), a także parametry dla rąk W i E (wskazujące na brak siły oraz układu na wejście do licytacji).

Wyjście: Wartości oczekiwane wyników dla kontraktów 2♠ oraz 4♠ rozgrywanych z ręki S. Wyniki zostały obliczone dla założeń “NS przed partią”.

Tabela 4.20

Kontrakt	Wartość oczekiwana wyniku
2♠	137.8125
4♠	213.1925

Wnioski: Wyniki przedstawione w tabeli 4.20 pozwalają stwierdzić, że w opisanym problemie bardziej opłacalną decyzją jest zalicytowanie 4♠.

4.2.3.2. Eksperyment “7”

Problem: *Po bloku przeciwnika 3♦ z ręką S1: ♠A7 ♥KD64 ♦A65 ♣K986 chcemy ustalić, co jest bardziej korzystne: licytacja (3BA) czy pas. Następnie chcemy sprawdzić co się zmieni, jeśli dysponujemy ręką S2: ♠A76 ♥KD64 ♦A5 ♣K986*[54]*.*

Cel: Na ogół blokujące otwarcie przeciwnika stawia drugą parę w trudnej sytuacji. Dlatego często gracz posiadający silniejszą rękę musi bardzo szybko ocenić, jaki kontrakt powinna jego strona w tym rozdaniu grać. W takim przypadku nie ma zbyt wiele miejsca na stosowanie ustaleń licytacyjnych, zamiast tego trzeba podjąć decyzję w oparciu o statystykę. Dwie podane ręce różnią się tylko tym, że jedna ma dwa piki i trzy kara, natomiast druga ma dwa kara i trzy piki.

53 Para grająca to rozdanie na linii NS stosowała bardziej agresywne otwarcia, stąd 11-14 zamiast 12-14 punktów.

54 Jedno z ustaleń dla tego typu sytuacji licytacyjnych zakłada, że z trójką pików należy dać kontrę. W przypadku gdy para NS stosuje to ustalenie, tak postawiony problem ma charakter czysto teoretyczny.

Opis: Do celów porównawczych zostaną przeprowadzone dwa eksperymenty. W ramach każdego z nich wygenerowanych zostanie 4000 rozdań. Gracz S ma jedną z podanych w hipotezie rąk, natomiast gracz E miał rękę zdefiniowaną jako blok na kolorze karowym (3-10 punktów, 6+ kar). Następnie dla obu rąk S zostaną obliczone i porównane wartości oczekiwane wyników dla kontraktu 3BA rozgrywanego z jego ręki, oraz dla kontraktu 3♦ rozgrywanego z ręki E.

Wejście: Liczba rozdań, ręce S1 i S2, parametry ręki N (3-10 punktów, 6+ kar).

Wyjście: Wartości oczekiwane wyników dla kontraktów 3BA rozgrywanych z rąk S1 i S2, oraz dla kontraktów 3♦ rozgrywanych z ręki E. Wyniki zostały obliczone dla założeń “obie przed partią”.

Tabela 4.21

Kontrakt	**Wartość oczekiwana wyniku**	
	Ręka S1	**Ręka S2**
3BA (S)	168.34	134.07
3♦ (E)	75.795	58.9125

Wnioski: Wyniki przedstawione w tabeli 4.21 pokazują, że strategia licytowania 3BA jest bardziej opłacalna niż pas, niezależnie od tego którą z rąk posiada S. Przewaga 3BA nad pasem jest bardziej wyraźna, gdy S posiada ♦A65 niż ♦A5. Faktycznie, zarówno z ręką S1, jak i S2, prawie żaden gracz nawet nie rozważy możliwości spasowania.

4.2.3.3 Eksperyment “8”

Problem: *Po licytacji przedstawionej w tabeli 4.22:*

Tabela 4.22

W	N	E	S
-	pas	1♦	kontra
pas	1♠	pas	2♣
pas	3♠	pas	?

dla gracza S z ręką: ♠4 ♥AK94 ♦K94 ♣AKD106 chcemy ustalić co jest bardziej korzystną decyzją: zalicytowanie 4♠, czy pas na 3♠. Założenia: „NS po partii”[55].

Cel: Trzeba ustalić, która z podanych decyzji przyniesie parze NS najbardziej korzystny rezultat, czyli policzyć wartość oczekiwaną dla kontraktów 3♠ i 4♠ z ręki N, oraz 3BA z ręki S (na potrzeby porównania odpowiedzi konkursowych w podrozdziale 5.2.1).

Opis: W ramach tego eksperymentu zostanie wygenerowanych 4000 rozdań, w których gracz S ma rękę podaną w treści problemu, natomiast gracze W, E i N posiadają ręce spełniające założenia licytacji z tabeli 4.22. Następnie zostanie wyznaczony rozkład lew do wzięcia przez N w piki.

Wejście: Liczba rozdań, podana ręka S, zestawy ograniczeń dla ręki W (0-4 punkty, mniej niż 4 kara), N (4-7 punktów, 6-7 pików) oraz E (12-17 punktów i układ na otwarcie 1♦, czyli 4+ kara).

Wyjście: Tabela 4.23

Tabela 4.23

Kontrakt	Wartość oczekiwana wyniku
4♠ (N)	210,875
3♠ (N)	110,62
3BA (S)	180,51

Wnioski: W tym przypadku licytacja (4♠) jest bardziej korzystną decyzją niż pas na 3♠, a różnica wartości oczekiwanych między tymi decyzjami wynosi około 100 punktów. Można również zauważyć, że wartość oczekiwana za 3♠ jest mniejsza niż 140 (tyle wynosi zapis za realizację tego kontraktu), co oznacza, że czasem nie uda się zrealizować nawet tego kontraktu.

55 Problem numer 6 z „Konkursu” 4/2016, *Brydż* 12/2015, s. 66.

5. Wnioski z eksperymentów

5.1. Weryfikacja i poszukiwanie praw brydżowych

Dzięki wprowadzeniu klasy *Experiment*, która zajmuje się generowaniem rozdań i obsługą programu rozgrywkowego, oraz zwraca gotowe statystyki (albo ich wybraną część), stało się możliwe bardzo szybkie przechodzenie od problemu decyzyjnego w licytacji do obliczania wartości oczekiwanych dla każdej z rozważanych przez użytkownika decyzji. Problem interpretacji licytacji w dalszym ciągu spoczywa na osobie przeprowadzającej eksperyment, ale teraz całe zadanie sprowadza się do wprowadzenia odpowiednich parametrów, oraz zbioru rozważanych decyzji licytacyjnych. Co prawda, czas działania programu do rozgrywki w otwarte karty nie pozwala na korzystanie z tego rodzaju usprawnień w trakcie gry na żywo (co jest w tym przypadku bardziej zaletą niż wadą, ponieważ utrudnia oszukiwanie), ale może być użytecznym narzędziem na potrzeby analizy prowadzonej po rozdaniu.

W rozdziale czwartym zostały zademonstrowane przykłady weryfikacji praw brydżowych (podrozdział 4.2.2) oraz problemów licytacyjnych (4.2.3). Wszystkie badane prawa brydżowe (które do tej pory oparte były na intuicji) zostały zweryfikowane pozytywnie, ponadto zostały odkryte kolejne prawa brydżowe (oparte na wcześniej zweryfikowanych). W tabeli 5.1 przedstawiona została lista zweryfikowanych hipotez.

Tabela 5.1

Treść hipotezy	**Numer eksperymentu**	**Podrozdział**
Jeżeli jeden z graczy ma wyraźnie silniejszą rękę od partnera, to bardziej korzystne jest rozgrywanie kontraktu z jego ręki.	1	4.2.2.1
Jeżeli para dysponuje dwoma równorzędnymi kolorami do gry, to najbardziej korzystna jest rozgrywka z ręki silniejszej, w kolor należący do ręki słabszej.	2	4.2.2.2
Rozgrywka w kolor rozłożony 4-4 jest bardziej opłacalna, niż w kolor 5-3, jeśli para ma takie kolory.	3	4.2.2.4
Rozgrywka w kolor rozłożony 4-4 jest bardziej opłacalna, niż w kolor 5-3, jeśli para ma takie kolory, nawet gdy kontrakt rozgrywany jest ze słabszej ręki.	5	4.2.2.5

Jeśli chodzi o rozwiązywanie problemów decyzyjnych w licytacji, to najlepszym sposobem ewaluacji będzie porównanie odpowiedzi na problemy z konkursu licytacyjnego miesięcznika „Brydż”, które powstały w oparciu o badania statystyczne, z intuicjami

(wspartymi wieloletnim doświadczeniem brydżowym) ekspertów. Wyniki tej konfrontacji zostaną zaprezentowane w podrozdziale 5.2.

5.2. Rozwiązywanie problemów decyzyjnych w licytacji

Konkurs „Brydża” to jedna z najstarszych rubryk ukazujących się na łamach miesięcznika „Brydż”. W ciągu roku odbywa się dziesięć edycji konkursu. W ramach każdej z nich czytelnicy (oraz eksperci) „Brydża” otrzymują do rozwiązania osiem problemów. Na ogół jest to siedem problemów licytacyjnych i jeden problem wistowy. Problemy licytacyjne należy rozwiązywać w oparciu o system licytacyjny „Wspólny Język” w wersji przedstawionej na stronie 67 każdego numeru „Brydża”.

Na podstawie odpowiedzi nadesłanych przez ekspertów oraz czytelników ustalana jest punktacja za poszczególne odpowiedzi, przy czym punkty są przyznawane tylko za te odpowiedzi, które zostały zaproponowane przez co najmniej jednego eksperta. Punktacja dla każdej z nich zależna jest od liczby ekspertów, którzy taką odpowiedź wybrali.

Przykład: *Punktacja za problem 4 z konkursu 6/2016 przedstawiona w tabeli 5.2*[56]:

Tabela 5.2

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
3♥	5	44	100
3BA	5	40	90
3♠	5	8	80
Pas	1	4	20

Ponieważ aż trzy odpowiedzi uzyskały poparcie takiej samej liczby ekspertów, kolejność decydująca o punktacji została ustalona w oparciu o głosy czytelników. Jak widać, odpowiedź 3♥ poparło o 4% czytelników więcej, niż 3BA, dlatego to za nią przyznano 100 punktów. Odpowiedź 3♠ poparło tylko 8% czytelników, ale decydujące znaczenie mają opinie ekspertów, dlatego była warta aż 80 punktów. Za to odpowiedź „Pas”, która została poparta tylko przez jednego eksperta i 4% czytelników, otrzymała znacznie mniej punktów – jedynie 20.

W podrozdziałach od 5.2.1 do 5.2.6 zostaną przedstawione rozwiązania problemów decyzyjnych pochodzących z różnych edycji „Konkursu”. Rozwiązania te będą oparte na wynikach eksperymentów. Uzyskane rezultaty zostaną zaprezentowane wraz z punktacją, oraz wybranymi odpowiedziami ekspertów.

56 „Konkurs 6/2016”, *Brydż* 4/2015, s. 61.

5.2.1. Problem 6 z konkursu 4/2016

Problem: *NS po partii, WE przed. Ręka S to ♠4 ♥AK94 ♦K94 ♣AKD106. Przebieg dotychczasowej licytacji w tabeli 5.3*[57]:

Tabela 5.3

W	N	E	S
-	pas	1♦	kontra
pas	1♠	pas	2♣
pas	3♠	pas	?

Ten problem został rozwiązany w ramach eksperymentu „8” (podrozdział 4.2.3.3). Z uzyskanych wyników wynikało, że zalicytowanie 4♠ jest bardziej korzystne, niż pas na 3♠. Punktację konkursową za ten problem przedstawiono w tabeli 5.4:

Tabela 5.4

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
3BA	8	64	100
4♠	6	24	80
pas	1	0	20

Z tabeli 5.4 wynika, że zarówno większość ekspertów, jak i czytelników wybrało odpowiedź, która nie została wzięta pod uwagę przy definiowaniu problemu[58]. Wynikało to prawdopodobnie z faktu dokonania błędnej interpretacji licytacji gracza zajmującego pozycję N. Sześciu ekspertów nie miało jednak problemów z interpretacją licytacji tego gracza, i wybierali 4♠: ***Henclik: 4♠****. Partner ma 6-7 pików z dwoma przegrywającymi (np. DW10xxxx) i koniec karty,* ***Jassem: 4♠****. Na 3BA widzę 8 lew i ani jednej więcej. Na 4♠ dostrzegam szansę, jeśli partner ma dobre 6 pików,* ***Wala: 4♠****. Dobry 6-kartowy kolor nieco podwiązany wystarcza. Trzyma bilans*[59]*.* Niestety komentarz eksperta, który wybrał odpowiedź „pas” nie został opublikowany na łamach „Brydża”. Jak widać, eksperci i czytelnicy, którzy poprawnie odczytali licytację N, wskazali odpowiedź najbardziej opłacalną z punktu widzenia statystyki.

57 Problem numer 6 z „Konkursu 4/2016”, *Brydż* 12/2015, s. 66.

58 Z tego powodu w tabeli 4.23 została przedstawiona również wartość oczekiwana wyniku za kontrakt 3BA, która była o około 30 punktów mniejsza od tej za 4♠.

59 „Konkurs 4/2016 – Omówienie wyników”, *Brydż* 2/2016, s. 62.

5.2.2. Problem 4 z konkursu 5/2016

Problem: *Obie po partii. Ręka S to ♠AD8643 ♥W ♦W ♣W9842. Przebieg dotychczasowej licytacji w tabeli 5.5:*

Tabela 5.5

W	N	E	S
-	pas	1♥	2♥
4♥	4♠	pas	pas
5♥	kontra	pas	?

Gracz S ma wybór pomiędzy zalicytowaniem 5♠, a spasowaniem. W przypadku, gdy wybierze 5♠, licytacja może zakończyć się także kontraktem 5♠ z kontrą, dlatego w przypadku rozpatrywania odzywki 5♠, należy policzyć także wartość oczekiwaną dla takiego kontraktu. W przypadku pasa, E będzie grał kontrakt 5♥ z kontrą. Wartości oczekiwane dla każdego z tych kontraktów (obliczone na próbie 4000 rozdań) zostały przedstawione w tabeli 5.6:

Tabela 5.6

Kontrakt	Wartość oczekiwana wyniku
5♠ (N)	-20.695
5♠ z kontrą (N)	-175.7875
5♥ z kontrą (E)	-272.975

Jak widać, najbardziej opłacalną decyzją jest zalicytowanie 5♠, ponieważ nawet gdy przeciwnicy skontrują, wartość oczekiwana wyniku będzie większa niż w przypadku, gdy WE zagrają 5♥ z kontrą. za ten problem zaprezentowano w tabeli 5.7:

Tabela 5.7

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
5♠	7	63	100
pas	7	37	90

Odpowiedzi ekspertów rozłożyły się równo, dlatego o punktacji zadecydowali czytelnicy – znaczna większość poparła odpowiedź 5♠ (czyli: licytuj). Prowadzący konkurs Marek Wójcicki podzielił odpowiadających na dwa „obozy”: optymistów (wybierających pasa) oraz pesymistów (wybierających 5♠). Jak się okazuje, przeprowadzanie skomplikowanej analizy nie było konieczne: ***Martens: 5♠****. Same odchylenia ujemne, jeżeli*

chodzi o kontrakt kierowy, ***Carruthers: 5♠****. Mój szósty pik i brak lew defensywnych sugerują, że pasowanie jest słabym pomysłem,* ***Krupiński: 5♠****. Partner jest po pasie, więc nie ma trzech lew, my możemy nie dołożyć żadnej, więc decyzja oczywista*[60]. Dla równowagi przytoczone zostały również dwie odpowiedzi „optymistów", których jednak w tym przypadku intuicja zawiodła: ***Henclik: pas****. Może te moje walety utworzą jakąś zacinkę i uda się bez jednej?,* ***Busse: pas****. Kontra oznacza dwa asy, a my mamy świetne informacje dla naszego partnera, bo przebijamy karo. Traktowanie kontry jako „do wyboru" bądź „mam coś w karcie" nie wytrzymuje krytyki*[61].

5.2.3. Problem 6 z konkursu 5/2016

Problem: *Obie po partii. Ręka S to ♠AW93 ♥K84 ♦3 ♣K10853. Przebieg dotychczasowej licytacji w tabeli 5.8:*

Tabela 5.8

W	N	E	S
1♦	1♠	kontra	2BA
3♦	3♠	4♦	4♠
pas	pas	5♦	?

Tym razem gracz S ma do wyboru pasa (który w tej sekwencji prowadzi do 5♠) lub kontrę. Oznacza to, że należy porównać wartości oczekiwane dla kontraktów 5♠ z ręki N oraz 5♦ z kontrą, z ręki W. Wyniki zostały przedstawione w tabeli 5.9:

Tabela 5.9

Kontrakt	Wartość oczekiwana wyniku
5♠ (N)	78.28
5♦ z kontrą (W)	622.175

Z tabeli 5.9 wynika, że kontra jest zdecydowanie bardziej opłacalna niż pas. W tabeli 5.10 przedstawiono punktację konkursową za ten problem:

Tabela 5.10

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
kontra	12	19	100
pas	2	48	50

60 „Konkurs 5/2016 – Omówienie wyników", *Brydż* 3/2016, s. 60.
61 Tamże.

Zdecydowana większość ekspertów nie miała problemu z wyborem bardziej korzystnej kontry, natomiast prawie połowa czytelników wybrała pasa. Znalezienie uzasadnienia dla kontry nie stanowiło trudności: ***Henclik: kontra**. Uważam, że pas byłby nieforsujący a nawet jeśli nie, to wygranie 5♠ jest mało realne. A mam trochę wartości defensywnych, więc oni powinni przegrać, **Gotard: kontra**. Chyba szybciej obłożymy ich na 5♦, niż sami wygramy kontrakt na wysokości pięciu, **Martens: kontra**. Może Splinter zamiast 1BA pomógłby teraz w podejmowaniu decyzji? Oni zapewne przegrywają, a my 5♠ raczej nie wygramy*[62].

Okazało się też, że dwa pasy dane przez ekspertów mogły wynikać z nieporozumienia: ***Krupiński: pas**. Najpierw dałem inwit, potem sam go przyjąłem, więc nie jesteśmy w pozycji z forsującym pasem*[63]. Po przytoczeniu tej odpowiedzi, prowadzący konkurs wyjaśnił też, że: *Ostatnio coraz większą popularność zdobywa podejście, iż do wykreowania pozycji forsującego pasa na wysokości czterech niezbędna jest wcześniejsza odzywka wyraźnie, jednoznacznie wskazująca, iż przewaga siły jest po naszej stronie*[64]. Oznacza to, to eksperci są na bieżąco z najnowszymi trendami w licytacji, podczas gdy czytelnicy pozostali trochę „z tyłu", co przy okazji bardzo dobrze mogłoby tłumaczyć tak duże poparcie dla odpowiedzi „pas" wśród czytelników.

5.2.4. Problem 3 z konkursu 6/2016

Problem: *Mecz. Obie po partii. Ręka S to ♠K ♥864 ♦K832 ♣Q8542. Przebieg dotychczasowej licytacji w tabeli 5.11:*

Tabela 5.11

W	N	E	S
-	1BA	pas	?

Gracz S w tej sytuacji ma podjąć decyzję, czy z posiadaną przez niego ręką powinien spasować, czy licytować (w tym przypadku 2BA). Ponieważ zapis za 1BA z nadróbką i za 2BA jest taki sam, należy uwzględnić, że 2BA daje możliwość zagrania 3BA, natomiast pas kończy licytację. Należy zatem wyznaczyć wartość oczekiwaną dla każdego z tych trzech kontraktów, aby ustalić czy dalsza licytacja jest bezpieczna (czyli 2BA daje wynik taki sam, lub minimalnie gorszy niż 1BA), oraz perspektywiczna (wartość oczekiwana wyniku za 3BA jest większa, niż za 1BA). Wyniki przedstawiono w tabeli 5.12:

62 „Konkurs 5/2016 – Omówienie wyników", *Brydż* 3/2016, s. 61-62.
63 Tamże, s62.
64 Tamże.

Tabela 5.12

Kontrakt	Wartość oczekiwana wyniku
1BA	73.855
2BA	3.74
3BA	-193.025

Na podstawie tabeli 5.12 nietrudno jest zauważyć, że nawet wartość oczekiwana wyniku za 1BA jest mniejsza, niż zapis za realizację tego kontraktu (90). W przypadku gdy grane będzie 2BA, wartość oczekiwana spada prawie do zera, natomiast za 3BA jest już ujemna (oczywiście tutaj uwzględniono, że partner ma 15-17 punktów, natomiast gdy ma mniej niż 17, to nie zalicytuje 3BA, tylko spasuje na 2BA). Wyniki te wskazują, że licytowanie 2BA jest najprawdopodobniej „drogą donikąd". W tabeli 5.13 przedstawiono punktację konkursową za ten problem:

Tabela 5.13

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
Pas	9	64	100
2BA	6	28	70
2♠	0	8	20

Większość ekspertów wybrała najbardziej opłacalnego pasa, podobnie jak prawie dwie trzecie czytelników. Eksperci uzasadniali wybór pasa w następujący sposób: ***Wala: pas****. Na inwit trochę za mało, zwłaszcza, że puste kolory. 8PC powinno wystarczyć do wygrania 1BA. Poszukiwanie lepszego kontraktu może być skuteczne, ale stanowi loterię,* ***Henclik: pas****. Niskie blotki i zła lokalizacja honorów są wskazówką jak postąpić w granicznej sytuacji,* ***Krupiński: pas****. Nie podoba mi się ta ręka. Niby 8PC, ale kolory mocno przewiewne. Jako, że moi partnerzy na otwarcie 1BA miewają częściej 14 niż 17, to nie dam nawet inwitu*[65].

Eksperci, którzy wybierali odpowiedź 2BA, tłumaczyli swoją odpowiedź tym, że grają w meczu: ***Cannell: 2BA****. Trudno ocenić wartość singlowego króla pik. Ale w meczu i po partii próbuję dograć końcówkę,* ***Nowosadzki: 2BA:*** *daję inwit bezatutowy. Karta jest obrzydliwa i w każdych innych warunkach pas, ale nie chcę stawiać 10 impów na prostej drodze. Trochę też zależy od sytuacji taktycznej*[66]. Na to, że niekoniecznie jest to problem licytacyjny, zwrócił uwagę też inny ekspert: **Martens:** *Problem bardzie taktyczny niż*

65 „Konkurs 6/2016 – Omówienie wyników", *Brydż* 4/2016, s. 59.
66 Tamże.

brydżowy. Zależy do stylu gry przeciwników i tego, jakie mamy oczekiwania co do wyniku meczu[67].

5.2.5. Problem 5 z konkursu 6/2016

Problem: *NS po partii, WE przed. Ręka S to ♠KW6432 ♥- ♦K73 ♣AD53. Przebieg dotychczasowej licytacji w tabeli 5.14*:

Tabela 5.14

W	N	E	S
pas	pas	1♣	1♠
2♦	pas	2♥	?

2♦ oznaczało transfer na kiery, a 2♥ siłę do 14 punktów.

Gracz S ma do wyboru pasa, kontrę lub zalicytowanie 2♠. Ponieważ kontra ma znaczenie wywoławcze (licytacja po niej wcale nie musi zakończyć się kontraktem 2♥ z kontrą), zostaną rozpatrzone tylko pas i 2♠. Aby ustalić, która z tych decyzji jest bardziej opłacalna, należy porównać wartości oczekiwane wyników za kontrakty 2♠ (z ręki S) oraz 2♥ (z ręki E). Wyniki zostały przedstawione w tabeli 5.15:

Tabela 5.15

Kontrakt	Wartość oczekiwana wyniku
2♥ (E)	-17.87
2♠ (S)	86.78

Z przeprowadzonego eksperymentu wynika, że wartość oczekiwana zapisu jest dodatnia tylko dla 2♠. W tabeli 5.16 przedstawiono punktację konkursową za ten problem:

Tabela 5.16

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
Kontra	9	16	100
2♠	4	64	70
pas	3	16	50

Wybór kontry eksperci uzasadniali w różny sposób: ***Henclik: kontra****. Pozostawia wszystkie drogi otwarte. Z solidną opozycją i misfitem pik, partner ukarni. W przeciwnym razie przeniesie na piki lub zgłosi swój kolor. Na alternatywne 2♠ mam za słaby kolor (niskie blotki) a 3♣ byłoby wyprawą w nieznane,* ***Carruthers: kontra.*** *Wygląda na*

67 Tamże.

oczywistą, ***Cannell: kontra.*** *Wywoławcza do kierów, modelowo coś w rodzaju układu 6-1-3-3*[68]*.*

Za odpowiedzią 2♠ eksperci również znaleźli przekonujące argumenty: ***Martens: 2♠****. Jeżeli mówimy o układzie to modelowo kontra, ale brak lew i trudny wist w razie ukarnienia. Dlatego decyduję się na 2♠,* ***Busse: 2♠****. Licytuję to, co mam w karcie: sześć pików z nadwyżką w sile, bo inaczej dałbym od razu 2♠ po 1♣. Nie daję kontry, bo nie mam ochoty grać w obronie 2♥, nie próbuję 3♣, gdy mam ich zaledwie cztery*[69]*.* Tylko najbardziej ostrożni eksperci wybrali pasa: ***Wala: pas****. Chciałoby się walczyć. Przeciwko powtórzeniu pików przemawia jednak brak podwiązań. Przeciwko kontrze – renons w kierach oraz brak dobrego wistu. No i w końcu nie taka duża siła, a jesteśmy po partii. Ponadto przypuszczam, że W jest górą limitowany jedynie uprzednim pasem,* ***Gartaganis: pas****. Partner jest po pasie, nie mógł podnieść pików, ma najmniej ze cztery kiery… Jesteśmy po partii, a mój kolor jest cieniutki. Pas wydaje się opcją ostrożną, ale racjonalną.*

5.2.6. Problem 7 z konkursu 4/2016

Problem: *Maksy. NS przed partią, WE po partii. Ręka S to ♠Q3 ♥W ♦AW763 ♣A10863. Przebieg dotychczasowej licytacji w tabeli 5.17*:

Tabela 5.17

W	N	E	S
-	-	-	1♦
1♥	kontra	1BA	2♣
2♥	kontra	pas	?

Pierwsza kontra N oznaczała od 7 punktów i co najmniej 4 piki, natomiast kontra w następnym okrążeniu była propozycyjna (więcej punktów niż minimum, brak wygodnej licytacji, możliwość obrony przeciwko 2♥[70]). Teraz S ma do podjęcia decyzję: spasować albo zalicytować 3♣ (zgłaszając w ten sposób układ 5-5 na kolorach młodszych). Aby ustalić, która z tych decyzji jest bardziej opłacalna, należy porównać wartości oczekiwane wyników dla kontraktów 2♥ z kontrą (z ręki W), oraz 3♣ (z ręki S). Wyniki zostały przedstawione w tabeli 5.18:

68 „Konkurs 6/2016 – Omówienie wyników”, *Brydż* 4/2016, s. 61.
69 Tamże.
70 Jeśli W ma co najmniej 5 kierów, a E co najwyżej trzy, to N może posiadać cztery lub więcej kart w tym kolorze.

Tabela 5.18

Kontrakt	Wartość oczekiwana wyniku
2♥ z kontrą (W)	81.84
3♣ (S)	54.57

Z tabeli 5.18 widać, że minimalnie bardziej opłacalną decyzją jest pas na 2♥ z kontrą, które najczęściej zakończy się wpadką (prawdopodobnie bez jednej). Natomiast 3♣ ma pewne szanse, ale biorąc pod uwagę, że wynik za 3♣ to 110, tak niska jego wartość oczekiwana wynika z tego, że najczęściej kontrakt ten zostanie przegrany. W tabeli 5.19 przedstawiono punktację konkursową za ten problem:

Tabela 5.19

Odzywka	Eksperci	Czytelnicy (%)	Punktacja
Pas	10	68	100
3♣	3	20	50
2♠	2	8	40

Punktacja konkursowa pokazuje, że i tym razem intuicja nie zawiodła ekspertów, ponieważ aż dziesięciu wybrało najbardziej opłacalnego pasa. Uzasadnienia dla wyboru takiej odpowiedzi nie były specjalnie skomplikowane: ***Jassem: pas****. Nie widzę powodu, aby kontra nie była karna – każdy już powiedział, co wiedział,* ***Henclik: pas****. Oczywisty – kontra jest karna, a przynajmniej mocno propozycyjna,* ***Cannell: pas****. Mam trochę defensywy – warto pokusić się o te magiczne +200 na maksy*[71]. Jeden z ekspertów, którzy wyłamali się, wybierając odpowiedź 3♣, argumentował to tak: ***Martens: 3♣****. Kontra partnera wskazuje na lepszą kartę i brak innej licytacji*[72].

5.3. Nowy sposób testowania losowości rozdań

Eksperymenty opisane w podrozdziale 4.2.1, które miały na celu sprawdzenie zgodności generowanych rozkładów (bez ograniczeń) z rozkładami matematycznymi, pozwoliły także odkryć nowy sposób testowania losowości rozdań. Uzyskane zostały dwie stałe, które odpowiadają średnim liczbom lew do wzięcia w kolor, oraz w bez atu.

Dzięki znajomości tych liczb można teraz sprawdzić losowość danego zbioru (najlepiej dużego) rozdań wygenerowanych przez jakiś program (albo rozdanych). W tym celu do obiektu klasy *Experiment* należy wprowadzić zestaw rozdań, a następnie je

71 „Konkurs 4/2016 – Omówienie wyników”, *Brydż* 2/2016, s. 62-63.
72 Tamże, s. 63.

rozegrać w celu uzyskania rozkładu lew. W oparciu o ten rozkład można będzie ustalić, jak bardzo rozdania odbiegały od zupełnie losowych, poprzez porównanie z dwiema stałymi.

Dla rozgrywki w kolor jest to liczba bliska 6,3 lewy, dla której proponuję nazwę: **stała kolorowa**, natomiast dla rozgrywki w bez atu jest to liczba bliska 6,0 lewy, dla której proponuję nazwę: **stała bezatutowa**.

Po uzyskaniu średniego rozkładu lew, należy przyrównać średnie liczby lew do wzięcia w kolor do stałej kolorowej, a średnie liczby lew do wzięcia w bez atu do stałej bezatutowej, obliczając odchylenie standardowe. Im większe odchylenie, tym mniej zbiór badanych rozdań odpowiada rozkładom matematycznym. Pewne anomalie można również zaobserwować bez obliczania odchylenia, np. że jedna linia statystycznie miała do wzięcia więcej lew w kolory starsze itp.

Przykład: Dla zbioru 200 rozdań dany jest rozkład lew przedstawiony w tabeli 5.1.

Tabela 5.20

Rozgrywający	♣	♦	♥	♠	BA
S	6.195	6.245	6.13	6.195	5.81
E	6.285	6.405	6.535	6.375	6.14
W	6.305	6.41	6.54	6.41	6.19
N	6.2	6.22	6.095	6.19	5.8

Przewagę karty na linii WE widzimy nawet bez znajomości wartości stałych. Średnia liczba lew do wzięcia w każdy kolor i w bez atu jest większa po stronie tej pary.

Dzięki znajomości stałej kolorowej oraz bezatutowej można zmierzyć wariancję oraz odchylenie standardowe dla danych przedstawionych w tabeli 5.20. Im większe będzie odchylenie, tym bardziej zbiór badanych rozdań odległy jest od rozkładu losowego. W tym celu stałą kolorową należy podstawić jako wartość oczekiwaną liczby lew do wzięcia w kolor: μ_K , natomiast stałą bezatutową jako wartość oczekiwaną liczby lew do wzięcia w BA: μ_{BA} . Odchylenie można policzyć dla jednego z graczy, np. S:

$$\sigma_S=\sqrt{((S_\clubsuit-\mu_K)^2+(S_\diamondsuit-\mu_K)^2+(S_\heartsuit-\mu_K)^2+(S_\spadesuit-\mu_K)^2+(S_{BA}-\mu_{BA})^2)}=0{,}30$$ [73]

albo dla wszystkich:

$$\sigma=\sqrt{(\sum_{X\in\{S,W,E,N\}}((X_\clubsuit-\mu_K)^2+(X_\diamondsuit-\mu_K)^2+(X_\heartsuit-\mu_K)^2+(X_\spadesuit-\mu_K)^2+(X_{BA}-\mu_{BA})^2))}=0{,}64$$.

Zatem odchylenie zbioru tych 200 rozdań od rozkładu losowego wynosi około 0,64 lewy.

73 Przyjęto dokładność do dwóch cyfr po przecinku.

6. Zakończenie

6.1. Znaczenie projektu

Przeprowadzone w ramach pracy eksperymenty pozwoliły na wykazanie prawdziwości obydwu postawionych we wstępie tez. Pierwszą z nich, która dotyczyła możliwości weryfikacji praw brydżowych w oparciu o badania statystyczne, potwierdziły eksperymenty od pierwszego do piątego, opisane w podrozdziale 4.2.2. Drugą tezę, dotyczącą możliwości rozwiązywania problemów decyzyjnych w licytacji za pomocą metod statystycznych, potwierdziły eksperymenty od szóstego do ósmego (opisane w podrozdziale 4.2.3), a także porównanie rozwiązań problemów konkursowych z odpowiedziami ekspertów (w podrozdziale 5.2). Ponadto, w ramach eksperymentu czwartego, zostały wyznaczone dwie stałe, których wartości wyznaczają średnią liczbę lew do wzięcia w bez atu (**stała bezatutowa**), oraz w kolor (**stała kolorowa**).

W ramach projektu powstał system prowadzenia badań statystycznych, na który składają się: generowanie rozdań z zadanymi ograniczeniami, oraz zbieranie rezultatów rozgrywki, w celu uzyskania danych statystycznych. System ten umożliwia weryfikację zarówno już istniejących praw brydżowych, jak i nowo sformułowanych (np. w oparciu o prawa zweryfikowane już wcześniej; taki charakter miał eksperyment piąty). Zamiast polegać tylko i wyłącznie na intuicji, można wykonać eksperyment, którego rezultaty (o ile wielkość próby nie będzie zbyt niska) pozwolą na weryfikację postawionej hipotezy.

Warto zauważyć, że możliwe jest też badanie już istniejących, lub zupełnie nowych konwencji licytacyjnych, które z założenia powinny być oparte na co najmniej jednym z praw brydżowych. Jeśli stosowanie konwencji A jest statystycznie bardziej opłacalne, niż stosowanie konwencji B, to znaczy, że ten poziom opłacalności da się zmierzyć i porównać właśnie za pomocą badań statystycznych. W takim przypadku badania te będą miały znacznie bardziej skomplikowany charakter, ponieważ z uwagi na potrzebę rozpatrzenia różnych sekwencji licytacyjnych konieczne będzie przeprowadzenie wielu niezależnych eksperymentów.

Jeśli chodzi o rozwiązywanie problemów decyzyjnych, to w trakcie gry w brydża na żywo nie będzie (i raczej nie powinno być) możliwe korzystanie ze wsparcia w postaci możliwości przeprowadzenia szybkiego eksperymentu, w celu wybrania najbardziej korzystnej odzywki licytacyjnej. Jednak w przypadku gry przez Internet (np. na bezpłatnej platformie Bridge Base Online) sytuacja wygląda inaczej – już w chwili obecnej,

przynajmniej część graczy dysponuje na tyle wydajnymi komputerami, że przeprowadzenie eksperymentu jeszcze w trakcie licytacji (oczywiście na niewielkiej próbie) może być dla pozostałych graczy niezauważalne. Można również w oparciu o wyniki eksperymentów rozwiązywać problemy licytacyjne z konkursu „Brydża".

Zastosowanie rozwiązań zawartych w projekcie w programach brydżowych, czyli dodanie możliwości formułowania i rozwiązywania problemów decyzyjnych w trakcie licytacji pozwoliłoby podnieść umiejętności licytacyjne graczy komputerowych, a co za tym idzie – poprawić ich skuteczność przy grze w brydża. Można to zrobić na dwa różne sposoby: pierwszym, a zarazem prostszym z nich jest po prostu przeprowadzanie eksperymentu w sytuacji, gdy sytuacja licytacyjna skomplikuje się do tego stopnia, że komputer nie jest w stanie posłużyć się zaprogramowanym systemem licytacyjnym, a po podjęciu decyzji „zapominałby" o eksperymencie. Drugi sposób polegałby na tym, że wszystkie wyniki uzyskane z eksperymentów byłyby zapisywane i przechowywane, a w razie potrzeby gracz komputerowy w pierwszej kolejności przeszukiwałby bazę eksperymentów, aby sprawdzić czy dany problem nie został już rozwiązany (albo jakiś podobny problem, przy czym to podobieństwo należałoby w odpowiedni sposób zdefiniować). Dopiero gdy okaże się, że jeszcze takiej wiedzy nie posiada, zostanie przeprowadzony nowy eksperyment i zapamiętanie zostaną jego rezultaty.

Z punktu widzenia części brydżystów wielką zaletą proponowanego rozwiązania może być możliwość rozstrzygania sporów pomiędzy partnerami. Często dochodzi do sytuacji, kiedy jeden z graczy podjął decyzję, która doprowadziła parę do słabego zapisu, a jego partner uważa, że ta decyzja była niepoprawna. Jednak zdarzają się sytuacje, w których decyzje poprawne z punktu widzenia statystyki mogą przynosić straty w pojedynczych rozdaniach. Jeśli gracze będą chcieli rozstrzygnąć spór tego rodzaju, to mogą najpierw wspólnie zinterpretować licytację, w celu sprawdzenia czy już na tym etapie nie pojawiły się rozbieżności (podjęta decyzja wcale nie musiała być błędna, lecz mogła być wynikiem nieporozumienia licytacyjnego). Następnie, jeśli żadne rozbieżności nie wystąpiły, mogą przeprowadzić eksperyment wprowadzając rękę gracza podejmującego decyzję, oraz wszystkie znane ograniczenia dla pozostałych rąk. W tym przypadku najlepiej będzie, jeśli próba będzie możliwie duża (np. 1000 rozdań). Z rezultatów takiego eksperymentu na ogół będzie można ustalić, czy to intuicja zawiodła gracza, czy też para trafiła na „pechowe" rozdanie.

6.2. Dalsze kierunki badań

Możliwość szybkiego i systematycznego poszerzania dostępnej wiedzy brydżowej przy użyciu eksperymentów wykonywanych przez komputery, pozwala w pewnym sensie na nowo otworzyć (i zdynamizować) badania nad teorią brydża.

Najbardziej oczywiste jest prowadzenie usystematyzowanych badań w celu skonstruowania nowych, oraz rozwijania już istniejących systemów licytacyjnych. Takie zmiany mogą mieć bardzo różny charakter: od drobnej korekty przedziału punktowego, z jakim dawana jest odzywka w konkretnej sekwencji, aż po odkrycie i zastosowanie w systemie zupełnie nowego (niekoniecznie opartego na intuicji) prawa, które wcale nie musi stanowić rewolucji na poziomie wspomnianych we wstępie „Systemów Słabych Otwarć". Wymagać to będzie usystematyzowanych i zakrojonych na szeroką skalę badań, najlepiej w wysokim stopniu zautomatyzowanych.

Można również zająć się badaniami nad tym, w jaki sposób w systemie licytacyjnym rozłożone są poszczególne zbiory informacji, np. czy są one rozłożone w sposób optymalny, oraz ustalaniem w jaki sposób należy je podzielić. Dobrym wstępem do takich badań jest pierwszy rozdział opracowania *Systemy Słabych Otwarć*, pt. *Co to jest dobry system*?[74] W tym kontekście można również pójść w kierunku całościowej analizy nowych i starych systemów – być może niektóre zapomniane już rozwiązania „odkryto by na nowo", natomiast wstępna analiza „nowego genialnego systemu" wykaże, że z punktu widzenia statystyki jest on pozbawiony sensu.

Jeśli dałoby się zautomatyzować badania, być może w dalszej perspektywie możliwa byłaby również automatyzacja konstruowania systemów licytacyjnych? Takie programy dysponowałyby bazą już odkrytych praw brydżowych, i po otrzymaniu określonych (przez człowieka) założeń generowałyby system licytacyjny. W chwili obecnej wydaje się to mało realne, jednak jest to ciekawy problem, który można próbować rozwiązać np. za pomocą algorytmów genetycznych. W razie potrzeby, gdyby program zabrnął na niezbadane jeszcze obszary teorii brydżowej (tzn. nie byłby w stanie odwołać się do żadnego z praw dostępnych w bazie), mógłby posługiwać się eksperymentami, w celu potwierdzenia lub odrzucenia samodzielnie stworzonych na potrzeby powstającego systemu licytacyjnego praw, bez żadnej interwencji ze strony programisty.

Zdecydowanie bardziej realistyczną alternatywą jest stworzenie programu do zarządzania systemem licytacyjnym, czyli takiego, w którym można by przechowywać, konstruować i modyfikować system licytacyjny, w sposób przystępny dla każdego

74 Stanisław Rumiński, *Systemy Słabych Otwarć*, Warszawa 1984, s. 10-22.

użytkownika nieznającego się na programowaniu (czyli posługując się graficznym interfejsem). Taki program w tle śledziłby wprowadzane zmiany i na bieżąco informował o ich konsekwencjach (np. dodanie jednego znaczenia do któregoś otwarcia powodowałoby aktualizację informacji na temat jego frekwencyjności). Taki „menedżer systemów licytacyjnych" mógłby również informować o wykrytych brakach w systemie, o duplikujących się sekwencjach (tzn. dwóch różnych sekwencjach, które oznaczają dokładnie to samo, a zatem jedna z nich może być zbędna), czy też innych nieścisłościach, a w razie potrzeby wspomagać użytkownika badaniami statystycznymi.

W chwili obecnej brydżowa sztuczna inteligencja radzi już sobie bardzo dobrze z rozgrywką. Z licytacją na ogół jest gorzej – w tej dziedzinie jest jeszcze możliwy dalszy postęp. Jednak jeśli weźmiemy pod uwagę, że licytacja jest po prostu formą komunikacji, to może okazać się, że najpoważniejszą barierą stojącą na przeszkodzie do poprawy jej skuteczności w licytacji, jest to, że komputery są uczone licytacji za pomocą systemów opracowanych przez ludzi. Wiele uproszczeń w tych systemach wynika z niedoskonałości ludzkiej pamięci – licytowanie systemem, którego nie da się zapamiętać, jest niepraktyczne. Sztuczna inteligencja jednak nie podlega takim ograniczeniom, zatem mogłaby się posługiwać dowolnie skomplikowanym systemem licytacyjnym, nie ryzykując przy tym żadnej pomyłki. Wynikałoby z tego, że o ile ludzie dobrze sobie radzą z tworzeniem systemów dla siebie, to może sztuczna inteligencja powinna również samodzielnie utworzyć system licytacyjny na swoje potrzeby. Dla ludzi może się on okazać skrajnie nieintuicyjny i mało zrozumiały, ale stosowany przez komputery może być bardziej skuteczny, niż licytowanie według „ludzkich reguł".

A. Bibliografia

1. Beling P., *Efektywne rozwiązanie problemu rozgrywki w otwarte karty w brydżu* [w] *Metody Informatyki Stosowanej 3/2011 (28)*, Szczecin 2011.

2. Jassem, K., *System licytacyjny Wspólny Język 2015*, Poznań 2015.

3. Jassem, P. , *Podejmowanie decyzji brydżowych za pomocą logiki rozmytej i metod regułowych*, Poznań 2014.

4. Perz P., *Algorytm generowania rozdań brydżowych z narzuconymi ograniczeniami*, Poznań 2009.

5. Rumiński S., *Systemy Słabych Otwarć*, Warszawa 1984.

6. Sobczyk M., *Statystyka*, Warszawa 2001.

7. Wójcicki M., *Standard „Brydża" Wspólny Język w XXI wieku*, Kraków 2013.

8. Miesięcznik *Brydż*.