Politechnika Wrocławska

Wydział Matematyki
Kierunek studiów: Matematyka stosowana
Specjalność: –

Praca dyplomowa – inżynierska

ZASTOSOWANIE STOCHASTYCZNEJ OPTYMALIZACJI DO GIER CZĘŚCIOWO OBSERWOWALNYCH

Aleksander Jakóbczyk

słowa kluczowe:
tutaj podajemy najważniejsze słowa kluczowe (łącznie nie powinny być dłuższe niż 150 znaków).

krótkie streszczenie:
Celem rozprawy jest wyznaczenie nieoczekiwanych strategii w grach częściowo obserwowalnych za pomocą metod stochastycznej optymalizacji. Praca będzie opierać się na wynikach Cauwet i Teytauda z 2018 roku, w których przedstawili nieoczekiwane strategie dla kilku klasycznych gier oraz kilku metod optymalizacji. Podjęta zostanie próba odtworzenia oraz rozszerzenia wyników na kolejną grę. Przeprowadzona zostanie analiza porównawcza dla różnych metod optymalizacji.

Opiekun pracy dyplomowej	dr inż. Andrzej Giniewicz	.....	.....
	Tytuł/stopień naukowy/imię i nazwisko	ocena	podpis

*Do celów archiwalnych pracę dyplomową zakwalifikowano do:**

a) kategorii A (akta wieczyste)

b) kategorii BE 50 (po 50 latach podlegające ekspertyzie)

** niepotrzebne skreślić*

pieczątka wydziałowa

Wrocław, rok 2022

Wrocław University
of Science and Technology

Faculty of Pure and Applied Mathematics
Field of study: Mathematics
Specialty: Theoretical Mathematics

Engineering Thesis

TYTUŁ PRACY DYPLOMOWEJ W JĘZYKU ANGIELSKIM

Aleksander Jakóbczyk

keywords:
tutaj podajemy najważniejsze słowa kluczowe w języku angielskim (łącznie nie powinny być dłuższe niż 150 znaków)

short summary:
Tutaj piszemy krótkie streszczenie pracy w języku angielskim (nie powinno być dłuższe niż 530 znaków).

Supervisor	dr inż. Andrzej Giniewicz	.....	.....
	Title/degree/name and surname	grade	signature

*For the purposes of archival thesis qualified to:**

a) category A (perpetual files)

b) category BE 50 (subject to expertise after 50 years)

** delete as appropriate*

stamp of the faculty

Wrocław, 2022

Spis treści

Wstęp **3**

1 Definicje, lematy, twierdzenia, przykłady i wnioski **5**
1.1 Typy gier 5
1.2 Definicje i Oznaczenia 6
1.2.1 Strategie proste 6
1.2.2 Strategie mieszane 6
1.3 Twierdzenia 7
1.4 Problem porównań wielokrotnych 8
1.5 Algorytmy wyścigowe 8
1.5.1 Algorytm Limited Empirical Bernstein Race (LEBR) 8
1.5.2 Improved LEBR (ILEBR) 9
1.5.3 Bernstein race without maximum race length 14

Podsumowanie **15**

Dodatek **17**

Wstęp

Rozdział 1

Definicje, lematy, twierdzenia, przykłady i wnioski

Celem algorytmów, których będziemy wykorzystywać, jest znalezienie optymalnej strategi w dwuosobowych grach częściowo obserwowalnych. Zdefiniujmy zatem podstawowe pojęcia potrzebne nam do tego, aby matematycznie opisać czym jest gra i czym jest strategia optymalna. W tym celu wprowadzimy kilka podstawowych definicji. Definicje dotyczące podstawy teorii gier pochodzą z [3],[4].

1.1 Typy gier

Podstawową kategorią, na jaką możemy podzielić gry, jest podział ze względu na czas, w którym gracze podejmują decyzje:

Definicja 1.1 (Gra w postaci strategicznej)**.** Jest to typ gry, w której gracze podejmują decyzje w tym samym momencie.

Definicja 1.2 (Gra w postaci ekstensywnej)**.** Jest to typ gry, w której gracze podejmują decyzje we wcześniej ustalonej kolejności.

Przykładami gier w postaci strategicznej są gry kamień papier nożyce, oszust czy też mora. Natomiast przykładami gier w postaci ekstensywnej są szachy, warcaby oraz go.

Gry możemy również dzielić ze względu na posiadaną wiedzę.

Definicja 1.3 (Gra z kompletną informacją)**.** Jest to typ gry, w której gracze mają informacje o możliwych przyszłych wynikach gry i o zbiorach możliwych strategii.

Definicja 1.4 (Gra częściowo obserwowalna)**.** Jest to przeciwnie gier z kompletną informacją.

Przykładami gier z kompletną informacją są szachy, warcaby oraz go. Natomiast przykładami gier częściowo obserwowanymi są wszelkie gry posiadające w rozgrywce pewien elementy losowe takie jak rzut kostką czy też dobieranie kart.

Istnieje jeszcze wiele innych podziałów gier ze wglądu na kategorie takie jak liczba graczy,zbiory dostępnych akcji, możliwość tworzenia koalicji i wiele innych.

1.2 Definicje i Oznaczenia

1.2.1 Strategie proste

Wprowadźmy podstawowe oznaczenia potrzebne nam do tego, aby móc zdefiniować czym jest strategia optymalna.

- $N = \{1, 2, \ldots, n\}$ – zbiór graczy,
- $A_i,\ i \in N$ – niepusty zbiór strategii czystych gracza i,
- $m_i = |A_i|$ – liczba strategi gracza i,
- $A = \prod_{i \in N} A_i$ – zbiór wszystkich strategii gry,
- $u_i : A \to \mathbb{R}$ – funkcja wypłaty gracza i,
- $a = (a_1, a_2, \ldots, a_n) = (a_i)_{i \in N},\ a_i \in A_i$ – profil gry w strategiach czystych,
- $u_i(a) = u_i(a, a_{-i})$ – wypłata gracza i z profilu a,
- $a_{-i} = (a_i)_{i \in N \setminus \{i\}}$, – profil wszystkich strategii poza strategią graca i.

Definicja 1.5 (Gra strategiczna)**.** Grą strategiczną nazywamy trójkę $GS = \langle N, (A_i)_{i \in N}, (u_i)_{i \in N} \rangle$.

Definicja 1.6 (Równowaga Nasha w strategiach czystych gry strategicznej)**.** Równowaga Nasha w strategiach czystych gry strategicznej jest to profil gry $a^* = (a_1^*, a_2^*, \ldots, a_N^*) \in A$, takim, że:

$$\forall\, i \in N\ \forall\, a_i \in A_i \quad u_i(a_i^*, a_{-i}^*) \geqslant u_i(a_i, a_{-i}^*)$$

Zatem jest to profil gry, w którym istnieje strategia czysta dająca nie gorsze wyniki od dowolnej innej strategii czystej. Okazuje się jednak, że taki stan nie zawsze istnieje w strategiach czystych, np. w grze kamień papier nożyce strategia grania tylko kamienia daje gorsze rezultat przeciwko graniu tylko papieru. Podobnie ze strategią grania tylko nożyc i grania tylko papieru.

1.2.2 Strategie mieszane

Definicja 1.7 (Strategia mieszana)**.** Strategia mieszana σ_i graca i w grze strategicznej $GS = \langle N, (A_i)_{i \in N}, (u_i)_{i \in N} \rangle$. Nazywamy rozkład prawdopodobieństwa na zbiorze strategi czystych A_i:

$$\sigma_i = (\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{im_i})$$

gdzie σ_{ik} oznacza prawdopodobieństwo, że gracz i zagra strategie czysta $k \in A_i$.

Fakt 1.8. *Strategia czysta jest szczególnym przypadkiem strategij mieszanej, w którym prawdopodobieństwo zagrania jednej z dostępnych strategii wynosi 1.*

Wprowadźmy dodatkowe oznaczenia:

- $\Sigma_i = \left\{ \sigma_i : A_i \to [0,1], \sum_{k=1}^{n} \sigma_{ik} = 1, \sigma_{ki} \geqslant 0 \right\}$ – zbiór strategii mieszanych gracza i,

- $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ – profil gry,
- $u_i(\sigma) = u_i(\sigma_i, \sigma_{-i})$ – wypłata gracza i z profilu σ,
- $\sigma_{-i} = (\sigma_i)_{i \in N \setminus \{i\}}$, – profil wszystkich strategii poza strategią graca i.

Definicja 1.9 (Równowaga Nasha w strategiach mieszanej gry strategicznej)**.** Profil gry strategicznej σ_i^* jest Równowagą Nasha gdy:

$$\forall\, i \in N \;\; \forall\, \sigma_i \in \Sigma_i \quad u_i(\sigma_i^*, \sigma_{-i}^*) \geqslant u_i(\sigma_i, \sigma_{-i}^*)$$

Równowaga Nasha interpretujemy jako taki profil gry, w którym żaden z graczy nie opłaca się zmieniać swojej strategi, ponieważ nie skutkuje to zwiększeniem swoich zysków.

1.3 Twierdzenia

Algorytmy 1, 2, 3, 4, 5 wykorzystywane w poniższej pracy oparte są o dwa twierdzenia a dokładniej o szczególne przypadki wynikające z nierówności 1.10 i 1.12:

Twierdzenie 1.10 (Nierówność Hoffdinga)**.** *Niech $X_1, X_2, \ldots, X_t$ będzie ciągiem niezależnych zmiennych losowych (i.i.d.) takim, że $a_i \leqslant X_i \leqslant b_i$, wtedy:*

$$S_t = \sum_{i=1}^{t} X_i, \quad c_i = b_i - a_i,$$
$$P(|S_t - \mathbb{E}(S_t)| \geqslant \epsilon) \leqslant 2 \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^{n} c_i^2}\right)$$

Lemat 1.11. *Niech $X_1, X_2, \ldots, X_t$ będzie ciągiem niezależnych zmiennych losowych (i.i.d.) takim, że $0 \leqslant X_i \leqslant 1$, wtedy:*

$$\overline{X}_t = \frac{S_t}{t}, \quad \mu = \mathbb{E}(X_i), \quad P(|\overline{X}_t - \mu| \leqslant \epsilon) = 1 - \delta,$$
$$\epsilon \leqslant \sqrt{\frac{\ln(2/\delta)}{2t}}$$

Twierdzenie 1.12 (Empiryczna nierówność Bernsteina)**.** *Niech $X_1, X_2, \ldots, X_t$ będzie ciągiem niezależnych zmiennych losowych (i.i.d.) takim, że $a \leqslant X_i \leqslant b$, wtedy:*

$$P(|\overline{X}_t - \mu| \geqslant \epsilon) \leqslant \delta, \quad \overline{\sigma}_t^2 = \frac{1}{t} \sum_{i=i}^{t} (X_i - \overline{X}_t)^2,$$
$$|\overline{X}_t - \mu| \leqslant \overline{\sigma}_t \sqrt{\frac{2\ln(3/\delta)}{t}} + \frac{3R \ln(3/\delta)}{t}$$

Lemat 1.13. *Niech $X_1, X_2, \ldots, X_t$ będzie ciągiem niezależnych zmiennych losowych (i.i.d.) takim, że $0 \leqslant X_i \leqslant 1$, wtedy:*

$$P(|\overline{X}_t - \mu| \leqslant \epsilon) = 1 - \delta,$$
$$\epsilon \leqslant \overline{\sigma}_t \sqrt{\frac{2\ln(3/\delta)}{t}} + \frac{3\ln(3/\delta)}{t}$$

1.4 Problem porównań wielokrotnych

Załóżmy, że z prawdopodobieństwem $1-\delta$ chcemy wiedzieć który z dwóch graczy, p_1 i p_2 jest lepszy. W tym celu będziemy przeprowadzać testy statystyczne, dla których prawdopodobieństwo pomyłki k-tego testu wynosi δ_k, aż do momentu, gdy jeden z graczy wygra przeważającą ilość razy. Wtedy po przeprowadzeniu n takich testów

$$P(\text{Chociaż jeden z } n \text{ testów się pomylił}) \overset{(*)}{\leqslant} \sum_{k=1}^{n} \delta_k \implies$$
$$P(\text{Żaden test się nie pomylił}) \leqslant 1 - \sum_{k=1}^{n} \delta_k,$$

gdzie nierówność oznaczona $(*)$ wynika z faktu, że $P(X+Y) \leqslant P(X) + P(Y)$.

Oznacza to, że musimy wprowadzić pewną korektę, aby ostateczne prawdopodobieństwo popełnienia błędu było mniejsze niż δ. Możemy zastosować jedna z dwóch poprawek:

1.4.1 Niech n będzie maksymalną liczbą testów jaką pozwalamy wykonać, aby wyznaczyć lepszego gracza. Wtedy $\delta_k = \frac{\delta}{n}$.

1.4.2 Niech δ_k spełnia nierówność $\delta \geqslant \sum_{k=1}^{\infty} \delta_k$. Wtedy niezależnie od ilości przeprowadzonych testów, ostateczne prawdopodobieństwo pomyłki będzie nie większe niż δ.

1.5 Algorytmy wyścigowe

Jednymi z algorytmów stosującymi dane korekty są algorytmy wyścigowe (Racing Algorytms). Dwoma najpopularniejszymi typami algorytmów racingowych są „Hoffding race" oraz „Bernstein race". Oparte są one odpowiednio o Twierdzenie 1.10 i Twierdzenie 1.13 oraz korektę 1.4.2 . Mają one jednak pewną wadę, mogą trwać one dużą ilość czasu, a w momencie, gdy poziom umiejętności porównywanych graczy jest sobie równy (prawdopodobieństwo wygranej wynosi 50%), wtedy z prawdopodobieństwem równym $1-\delta$ algorytm się nigdy nie zatrzyma.

W celu poradzenia sobie z problemami, jakie wiążą się z klasycznymi algorytmami wyścigowymi, wprowadzimy tzw. Limited Racing algorytm. Załóżmy zatem dodatkowy warunek, który mówi, że przerywamy działanie algorytmu w momencie, gdy empiryczna wartość oczekiwana z prawdopodobieństwem większym bądź równym $1-\delta$ jest znana z dokładnością co do zadanego ϵ. W poniższej pracy będziemy przyjmować $\epsilon = 0.01$ oraz $\delta = 0.05$.

1.5.1 Algorytm Limited Empirical Bernstein Race (LEBR)

Klasyczny algorytm racingowy opiera się na szeregu ϵ_t który posiada właściwość że zdarzenie $\mathcal{E} = \{|\overline{X}_t - \mu| \leqslant \epsilon_t,\ t \in \mathbb{N}^+\}$ występuje z prawdopodobieństwem nie mniejszym niż $1-\delta$. Niech δ_k będzie dodatnim szeregiem spełniającym $\delta \geqslant \sum_{k=1}^{\infty} \delta_k$ wtedy korzystając z Lematu 1.13

$$\epsilon_t = \overline{\sigma}_t \sqrt{\frac{2\ln(3/\delta_t)}{t}} + \frac{3\ln(3/\delta_t)}{t}$$

Ponieważ δ_k sumuje się co najmniej do δ oraz $(\overline{X}_t - \epsilon_t, \overline{X}_t + \epsilon_t)$ jest przedziałem ufności dla μ o współczynniku ufności $1-\delta_t$ otrzymanym z Lematu 1.13, uzyskujemy że zdarzenie $\mathcal{E}$

występuję z prawdopodobieństw nie mniejszym niż $1-\delta$. Identyczny rezultat otrzymujemy stosując ϵ_t oparte o Lemat 1.11 jednak zmienia się wtedy postać ϵ_t. W pracy [1] algorytm opierał się o szereg $\delta_t = \frac{c\delta_t}{t^2},\ c = \frac{6}{\pi^2}$. Pseudo kod jest pokazany jako Algorytm 1 omówmy zarys metody. Rozgrywamy t gier, i wyznaczamy granice górna $\mathrm{UB} = \min\limits_{1 \leqslant k \leqslant t}(\overline{X}_k + c_k)$ oraz granice dolną $\mathrm{LB} = \max(0, \max\limits_{1 \leqslant k \leqslant t}(\overline{X}_k - c_k))$. Algorytm kończy się w momencie gdy $\mathrm{UB} - \mathrm{LB} < 2\epsilon$. Otrzymane w ten sposób $\overline{X}$ z prawdopodobieństwem nie mniejszym niż $1-\delta$ wynosi μ z dokładnością co zadanego ϵ.

Algorithm 1 LEBR

Ensure: precision ϵ, probabilitys δ_k
  $LB \leftarrow 0, \quad UB \leftarrow \infty, \quad t \leftarrow 0, \quad n \leftarrow 1$
  while $UB - LB > 2\epsilon$ **do**
    $t \leftarrow t + 1$
    Obtain X_t
    $\epsilon_n \leftarrow \overline{\sigma}_n \sqrt{\frac{2\ln(3/\delta_n)}{n}} + \frac{3\ln(3/\delta_n)}{n}$
    $LB \leftarrow \max(LB, \overline{X_n} - \epsilon_n)$
    $UB \leftarrow \min(BU, \overline{X_n} + \epsilon_n)$
    $n \leftarrow n + 1$
  end while
  return $\overline{X_t}$

1.5.2 Improved LEBR (ILEBR)

Algorytm 1 opiera się na korekcie 1.4.2 która zakłada możliwie nieskończoną ilość testów. Jednak dołożenie ograniczenia na temat żądanej dokładności rzędu ϵ pozwala nam znaleźć maksymalną ilość testów jaką należy wykonać aby z prawdopodobieństwem nie mniejszym niż $1-\delta$ empiryczna wartość oczekiwana była równa teoretycznej wartości oczekiwanej z dokładnością co do ϵ. Niech $\delta_k = \frac{\delta}{n_{max}}$ gdzie n_{max} oznacza maksymalną potrzebną ilość testów. Z Lematu 1.11 niezależnie od odchylenia standardowego zmiennej losowej X_i jesteśmy w stanie oszacować n_{max} rozwiązując analitycznie poniższe równie 1.1

$$\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2n_{max}}} \tag{1.1}$$

Dla $\epsilon = 0.01$ i $\delta = 0.05$ rozwiązaniem numerycznym równania 1.1 jest $n_{max} = 74539.85$. Oznacza to że maksymalna ilość gier jaką należy rozegrać między dwoma graczami aby z prawdopodobieństwem nie mniejszym niż $1-0.05$ oszacować prawdopodobieństwo wygrania gry przez pierwszego gracza z dokładniejsi co do 0.01 wynosi $\lceil n_{max} \rceil = 74540$.

Podobnie możemy postąpić w przypadku nierówności Bernsteina. Na początku jednak musimy oszacować z góry $\overline{\sigma}_t$. Przyjmijmy, że X_i jest zmienną z rozkładu zero-jedynkowego. Wtedy maksymalna możliwa wariancja dla takiej zmiennej losowej wynosi $\sigma^2 = 0.5$. Jest to również maksymalna możliwa wartość dla naszej empirycznej wariancji. Podstawiając wtedy $\delta_n = \frac{0.05}{n}, \epsilon = 0.01$ do Lematu 1.13 otrzymujemy:

$$0.01 \leqslant \sqrt{\frac{\ln(\frac{3n_{max}}{0.05})}{n_{max}}} + \frac{3\ln(\frac{3n_{max}}{0.05})}{n_{max}} \tag{1.2}$$

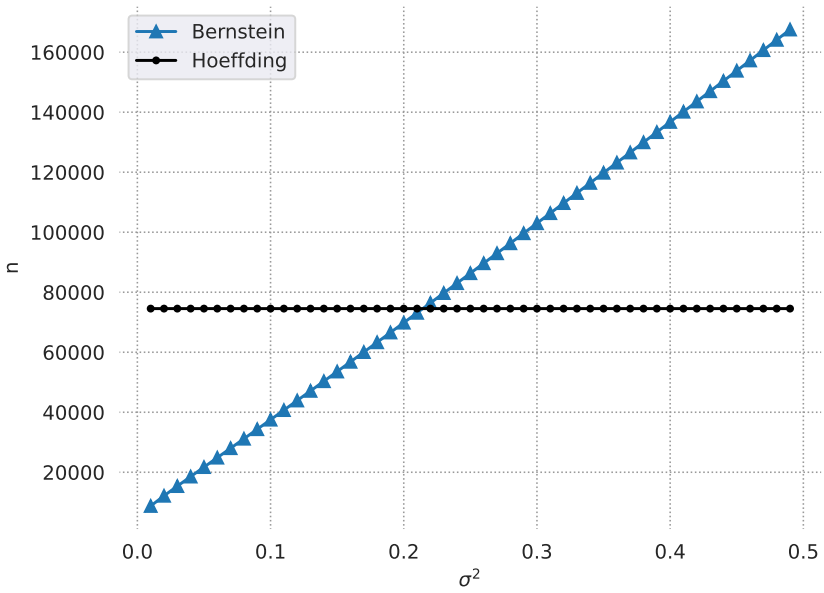

Figure 1.1: Wykres maksymalnej potrzebnej ilości testów w zależności od wariancji zmiennych losowych X_i w przypadku gdy liczba testów jest równa liczbie gier ($t = n$) dla $\epsilon = 0.01$ i $\delta = 0.05$.

Rozwiązaniem numerycznym równania 1.2 jest $n_{max} = 170986$. Jednak granice oparte o nierówność Bernsteina zalezą od wariancji. Sprawdźmy zatem jak wygląda n_{max} w zalotności od σ^2. Z Rysunku 1.1 widzimy zatem, że algorytm oparty o nierówność Bernsteina radzi sobie lepiej w przypadku gdy niskiej wariancji. Co więcej niezależnie od ilości wykonywanych testów, dla $\delta_n = \frac{0.05}{74540}$ maksymalna ilość gier po której z prawdopodobieństwem $1 - 0.05$ wiemy że $|\overline{X_i} - \mu| \leqslant 0.01$ wynosi 74540. Oznacza to że do Algorytmu 1 możemy zastosować funkcje ϵ_k opartą o $\delta_k = \frac{\delta}{n_{max}}$ oraz dołożyć nowy warunek zatrzymania algorytmu gdy liczba wykonanych testów przekroczy n_{max}. Dotychczas uwzględnione poprawki przedstawione sa przy Algorytmu 2.

Algorithm 2 ILEBR 1

Ensure: precision ϵ, probability δ
  $LB \leftarrow 0, \quad UB \leftarrow \infty, \quad t \leftarrow 0, \quad n \leftarrow 1$
  Find n_{max} such as $\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2n_{max}}}$
  $\delta_n = \delta / n_{max}$
  while $UB - LB > 2\epsilon$ or $n < n_{max} + 1$ **do**
    $t \leftarrow t + 1$
    Obtain X_t
    $\epsilon_n \leftarrow \overline{\sigma}_n \sqrt{\frac{2\ln(3/\delta_n)}{n}} + \frac{3\ln(3/\delta_n)}{n}$
    $LB \leftarrow \max(LB, \overline{X_n} - \epsilon_n)$
    $UB \leftarrow \min(BU, \overline{X_n} + \epsilon_n)$
    $n \leftarrow n + 1$
  end while
  return $\overline{X_t}$

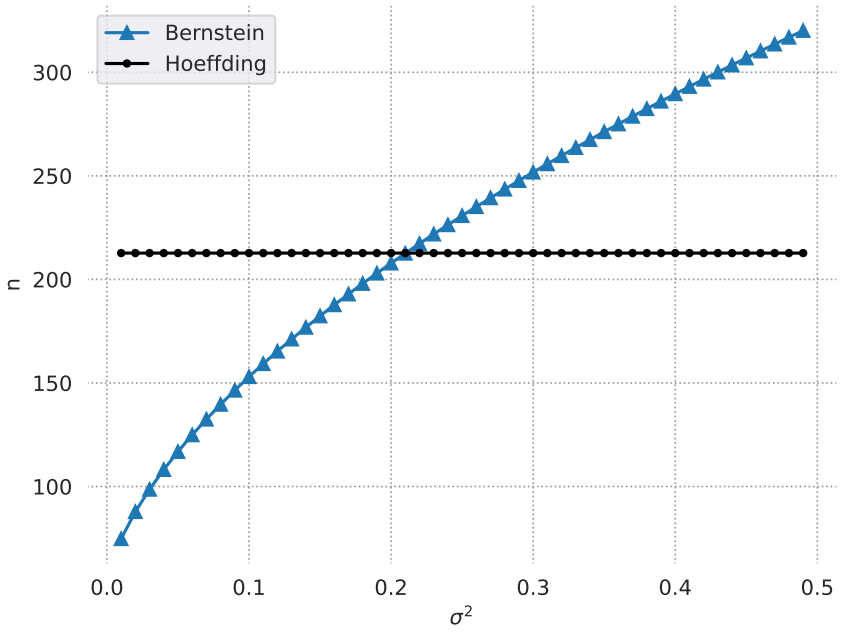

Figure 1.2: Wykres maksymalnej potrzebnej ilości testów w zależności od wariancji zmiennych losowych X_i w przypadku gdy liczba rozegranych gier jest równa liczbie przeprowadzonych testów do kwadratu ($t = n^2$) dla $\epsilon = 0.01$ i $\delta = 0.05$.

Algorytmy 1 i 2 przeprowadzały testy po każdej rozegranej grzęz, skutkuje to wytopieniem bardzkiej ilości testów. Zamiast przeprowadzać test po każdej rozegranej grze, niech test odbywa się w momencie gdy liczba rozegranych gier będzie równa pewnej funkcji zależnej od ilości przeprowadzonych testów. Na podstawie artykuł [2] wiemy że funkcja która pozwoli nam na polepszenie wyników jest $t = n^2$. Przeprowadźmy identyczną procedure jak w przypadku równań 1.1 i 1.2. Tym razem jednak zamiast stosować $t = n$ do Lematów 1.11 i 1.13 zastosujmy $t = n^2$.

W przypadku granicy opartej o nierówność Hoffdina otrzymujemy

$$\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2n_{max}^2}} \implies t_{max} = \lceil n_{max} \rceil^2 = \lceil 213 \rceil^2 = 45369. \tag{1.3}$$

W przypadku granicy opartej o nierówność Bernsteina otrzymujemy

$$0.01 \leqslant \sqrt{\frac{\ln(\frac{3n_{max}}{0.05})}{n_{max}^2} + \frac{3\ln(\frac{3n_{max}}{0.05})^2}{n_{max}}} \implies t_{max} = \lceil n_{max} \rceil^2 = \lceil 324^2 \rceil = 104976. \tag{1.4}$$

Porównując ze sobą uzyskane wartość z wzoru 1.1 oraz 1.3 widzimy że udało się zmniejszyć maksymalną ilość gier jakie należny wykonać z 74540 do 45369. Wprowadźmy zatem nowo uzyskane poprawki trwożąc Algorytm 3.

Algorithm 3 ILEBR 2

Ensure: precision ϵ, probability δ
  $LB \leftarrow 0, \quad UB \leftarrow \infty, \quad t \leftarrow 0, \quad n \leftarrow 1$
  Find n_{max} such as $\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2n_{max}^2}}$
  $\delta_n = \delta / n_{max}$
  while $UB - LB > 2\epsilon$ or $n < n_{max} + 1$ **do**
    repeat
      $t \leftarrow t + 1$
      Obtain X_t
    until $t = n^2$
    $\epsilon_n \leftarrow \overline{\sigma}_n \sqrt{\frac{2\ln(3/\delta_n)}{n^2}} + \frac{3\ln(3/\delta_n)}{n^2}$
    $LB \leftarrow \max(LB, \overline{X_n} - \epsilon_n)$
    $UB \leftarrow \min(BU, \overline{X_n} + \epsilon_n)$
    $n \leftarrow n + 1$
  end while
  return $\overline{X_t}$

Do tej pory Wszystkie stosowane algorytmy wyznaczały nam prawdopodobieństwo wygranej pierwszego gracza. Jednak nam nie zależny na tym dokładnie znać prawdopodobieństwo wygranej graczy. Głównym celem algorytmów jest odnalezienie lepszego gracza. Pozwala nam to modyfikacje Algorytmu 3 o nowe warunki zatrzymania.

Algorithm 4: ILEBR* 1

Ensure: precision ϵ, probability δ
  $LB \leftarrow 0, \quad UB \leftarrow \infty, \quad t \leftarrow 1, \quad n \leftarrow 0$
  Find n_{max} such as $\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2n_{max}^2}}$
  $\delta_n = \delta / n_{max}$
  while $(UB - LB > 2\epsilon$ or $n < n_{max} + 1)$ and $(BU > 0.5$ or $LB < 0.5)$ **do**
    repeat
      $t \leftarrow t + 1$
      Obtain X_t
    until $t = n^2$
    $\epsilon_n \leftarrow \overline{\sigma}_n \sqrt{\frac{2\ln(3/\delta_n)}{n^2}} + \frac{3\ln(3/\delta_n)}{n^2}$
    $LB \leftarrow \max(LB, \overline{X_n} - \epsilon_n)$
    $UB \leftarrow \min(BU, \overline{X_n} + \epsilon_n)$
    $n \leftarrow n + 1$
  end while
  if $LB > 0.5$ **then**
    return p_1 win
  else if $UB < 0.5$ **then**
    return p_2 win
  else if $\overline{X_n} > 0.5$ **then**
    return p_1 win
  else
    return p_2 win
  end if

Do tej pory staraliśmy się ograniczyć z góry maksymalną ilość rozgrywanych gier. Jednak oczywistym faktem jest że nie ma sensu testowania który z graczy jest lepszy w monecie gdy ilość rozegranych gier jest niewystarczająco mały. Jadnak jak wyznaczyć nasze minimalne t po którym zaczniemy testować? Algorytm oparte o nierówność Bernsteina najszybciej wyznaczają wynik w przypadku gdy wariancja naszej zmiennej losowej wynosi 0 (jeden z graczy zawsze wygrywa). Dla Algorytmu **??** interesuje nas monet od którego będziemy w stanie rozróżnić kiedy dolna bać górna granica przekroczy $\frac{1}{2}$. Zatem oszacujmy n_{min} korzystać z Lematu 1.13 dla $\overline{\sigma}_t = 0$ i $t = n^2$.

$$0.5 \leqslant \frac{3\ln(\frac{3n_{min}}{0.05})}{n_{min}} \implies {}^2t_{min} = \lceil n_{min} \rceil^2 = \lceil 5.93749 \rceil^2 = 36.$$

Oznacza to, że przy $\epsilon = 0.01$ i $\delta = 0.05$ w przypadku gdy jeden graczy wygrywa zawsze, będziemy w stanie to stwierdzić nie prędzej niż po 36 grach. Na tej podstawie wyznaczy nowe n_{max} uwzględniając pomijanie pierwsze niepotrzebne testy.

$$0.01 = \sqrt{\frac{\ln(2n_{max}/0.05)}{2(n+6)^2_{max}}} \implies t_{max} = \lceil n_{max} + 6 \rceil^2 = \lceil 207 + 6 \rceil^2 = 45369. \tag{1.5}$$

Chociaż równanie 1.5 nie zmniejszyło nam maksymalnej liczby testów jakie należy wykonać to pozwoliło nam ono na zmniejszenie ϵ_n. Oznacza to że stosują odroczenie pierwszych testów, pozwala nam na dokładniejsze oszacowanie naszej górnej i dolnej granicy w stosowanych algorytmach.

Algorithm 5: ILEBR* 2

Ensure: precision ϵ, probability δ
  $LB \leftarrow 0, \quad UB \leftarrow \infty, \quad t \leftarrow 1, \quad n \leftarrow 0$
  Find n_{max} such as $\epsilon = \sqrt{\frac{\ln(2n_{max}/\delta)}{2(n+6)^2_{max}}}$
  $\delta_n = \delta / n_{max}$
  while $(UB - LB > 2\epsilon$ or $n < n_{max} + 1)$ and $(BU > 0.5$ or $LB < 0.5)$ **do**
    repeat
      $t \leftarrow t + 1$
      Obtain X_t
    until $t = n^2$
    $\epsilon_n \leftarrow \overline{\sigma}_n \sqrt{\frac{2\ln(3/\delta_n)}{(n+6)^2}} + \frac{3\ln(3/\delta_n)}{(n+6)^2}$
    $LB \leftarrow \max(LB, \overline{X_n} - \epsilon_n)$
    $UB \leftarrow \min(BU, \overline{X_n} + \epsilon_n)$
    $n \leftarrow n + 1$
  end while
  if $LB > 0.5$ **then**
    return p_1 win
  else if $UB < 0.5$ **then**
    return p_2 win
  else if $\overline{X_n} > 0.5$ **then**
    return p_1 win
  else
    return p_2 win
  end if

1.5.3 Bernstein race without maximum race length

Wykorzystując Lemat 1.13 oraz korektę 1.4.2 z $\delta_k = \frac{c\delta}{k^2}, c = \frac{6}{\pi^2}$ otrzymujemy, że dla ciągu $X_1, X_2, \ldots, X_t$ i.i.d. zmiennych losowych takim, że $0 \leqslant X_i \leqslant 1$

$$\epsilon_{t,k} \leqslant \overline{\sigma}_t \sqrt{\frac{2\ln(3/\delta_k)}{t}} + \frac{3\ln(3/\delta_k)}{t} = \overline{\sigma}_t \sqrt{\frac{2\ln(\frac{k^2\pi^2}{2\delta})}{t}} + \frac{3\ln(\frac{k^2\pi^2}{2\delta})}{t}$$

Wtedy $e_{t,k}$ interpretujemy jako maksymalną różnicę między empiryczną a teoretyczną wartością oczekiwaną po przeprowadzeniu k testów i rozegraniu t gier, z prawdopodobieństwem pomyłki równym

Lemat 1.14. *Niech liczba rozegranych gier będzie funkcja zależną od k ($f(k) = t$) oraz niech $\delta_k = \frac{\delta}{g(k)}$ gdzie $\delta \geqslant \sum_{k=1}^{\infty} \frac{\delta}{g(k)}$ i $\ln(g(k)) \in o(f(k))$. Wtedy $\lim_{k\to\infty} e_{f(k),k} = 0$*

Dowód Faktu 1.14.

$$0 \leqslant X_i \leqslant 1 \implies \overline{\sigma}_t^2 \leqslant \frac{1}{2}$$

$$\epsilon_{f(k),k} \leqslant \sqrt{\frac{\ln(\frac{3g(k)}{\delta})}{f(k)}} + \frac{3\ln(\frac{3g(k)}{\delta})}{f(k)}$$

Z założeń wiemy ze $ln(g(k)) \in o(f(k))$, zatem

$$\lim_{k\to\infty} \frac{\ln(\frac{3f(k)}{\delta})}{f(k)} = 0$$

Co ostatecznie z tw. o trzech ciągach daje nam

$$0 \leqslant \lim_{k\to\infty} e_{f(k),k} \leqslant 0 \implies \lim_{t\to\infty} e_{f(k),k} = 0$$

□

Lemat 1.14 pozwala nam na to aby ograniczyć ilość przeprowadzanych testów. Ponieważ podejście oparte o testowane który z graczy jest lepszy po każdej rozegranej grze prowadzi do nadmiarowej ilości wykonywanych testów. Przykładem funkcjami jakimi użyć do Lematu 1.14 są $f(k) = k^2$, $g(k) = \frac{6/\pi^2}{k^2}$. Teaki dobór funkcji oznacza ze k-ty test odbywa się gdy liczba przeprowadzonych gier wynosi k^2.

Podsumowanie

Podsumowanie w pracach matematycznych nie jest obligatoryjne. Warto jednak na zakończenie krótko napisać, co udało nam się zrobić w pracy, a czasem także o tym, czego nie udało się zrobić.

Dodatek

Dodatek w pracach matematycznych również nie jest wymagany. Można w nim przedstawić np. jakiś dłuższy dowód, który z pewnych przyczyn pominęliśmy we właściwej części pracy lub (np. w przypadku prac statystycznych) umieścić dane, które analizowaliśmy.

[1]

Bibliography

[1] Cauwet, M.-L., Teytaud, O. Surprising strategies obtained by stochastic optimization in partially observable games. In *2018 IEEE Congress on Evolutionary Computation (CEC)* (2018), IEEE, pp. 1–8.

[2] Heidrich-Meisner, V., Igel, C. Non-linearly increasing resampling in racing algorithms. In *European Symposium on Artificial Neural Networks* (2011), Evere, Belgium: d-side publications, pp. 465–470.

[3] Płatkowski, T. Wstęp do teorii gier. *Uniwersytet Warszawski* (2012).

[4] Prisner, E. *Game theory through examples*, vol. 46. American Mathematical Soc., 2014.