

**AKADEMIA GÓRNICZO-HUTNICZA  
IM. STANISŁAWA STASZICA W KRAKOWIE**

---

Wydział Informatyki, Elektroniki i Telekomunikacji  
Katedra Informatyki



**PROJEKT INŻYNIERSKI**

**SYSTEM OBLICZAJĄCY WYNIKI WYBORÓW DLA  
UOGÓLNIENIA SYSTEMU K-BORDA**

**TOMASZ KASPRZYK, DANIEL OGIELA  
JAKUB STĘPAK**

OPIEKUN:  
dr hab. inż. Piotr Faliszewski

---

Kraków 2016

## **OŚWIADCZENIE AUTORÓW PRACY**

OŚWIADCZAMY, ŚWIADOMI ODPOWIEDZIALNOŚCI KARNEJ ZA POŚWIADCZENIE NIEPRAWDY, ŻE NINIEJSZY PROJEKT WYKONALIŚMY OSOBIŚCIE I SAMODZIELNIE W ZAKRESIE OPISANYM W DALSZEJ CZĘŚCI DOKUMENTU I ŻE NIE KORZYSTALIŚMY ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W DALSZEJ CZĘŚCI DOKUMENTU.

.....  
PODPIS

.....  
PODPIS

.....  
PODPIS

## Spis treści

<b>1</b>	<b>Wizja produktu</b>	<b>5</b>
1.1	Opis problemu . . . . .	5
1.2	Możliwości . . . . .	7
1.3	Motywacja do stworzenia projektu . . . . .	7
1.4	Użytkownicy . . . . .	8
1.5	Usługi dostarczanie przez system . . . . .	8
1.5.1	Usługi wymagane . . . . .	8
1.5.2	Usługi pożądane . . . . .	8
1.5.3	Usługi dodatkowe . . . . .	8
<b>2</b>	<b>Studium wykonalności</b>	<b>8</b>
2.1	Opis wymagań . . . . .	8
2.1.1	Wymagania funkcjonalne . . . . .	8
2.1.2	Wymagania niefunkcjonalne . . . . .	11
2.2	Strategia testowania . . . . .	11
2.3	Aspekt technologiczny . . . . .	12
2.4	Analiza ryzyka . . . . .	12
<b>3</b>	<b>Wybrane aspekty realizacji</b>	<b>12</b>
3.1	Lista modułów . . . . .	12
3.2	Diagram komunikacji . . . . .	13
3.2.1	Opis diagramu . . . . .	14
<b>4</b>	<b>Organizacja pracy</b>	<b>14</b>
4.1	Metodyka pracy . . . . .	14
4.2	Podział prac . . . . .	14
4.3	Narzędzia do zarządzania projektem . . . . .	15
4.4	Opis kolejnych wersji systemu . . . . .	15
4.4.1	Wersja 1. . . . .	15
4.4.2	Wersja 2. . . . .	15
4.4.3	Wersja 3. . . . .	15
4.4.4	Wersja 4 - końcowa. . . . .	16
<b>5</b>	<b>Opisy algorytmów heurystycznych</b>	<b>17</b>
5.1	Algorytm zachłanny . . . . .	17
5.2	Algorytm zachłanny według zasady Chamberlina-Couranta . . . . .	17
5.3	Algorytm genetyczny . . . . .	17
<b>6</b>	<b>Efekt końcowy</b>	<b>18</b>
6.1	Oczekiwane rezultaty . . . . .	18
6.2	Prezentacja i jakość wyników . . . . .	19
6.2.1	Prezentacja tabeli z wszystkimi wynikami dla danych wyborów . . . . .	19
6.2.2	Prezentacja szczegółowych wyników wyborów . . . . .	20

<b>7</b>	<b>Testy porównawcze</b>	<b>22</b>
7.1	Zrównanie jakości działania algorytmów zachłannych . . . . .	22
7.1.1	Przewidywany schemat działania algorytmów zachłannych . . . . .	22
7.1.2	Wartość graniczna parametru $p$ . . . . .	22
7.1.3	Stopień pokrywania się zwycięzców dla różnych wartości parametru $p$ . . . . .	22
7.2	Porównanie algorytmu genetycznego i algorytmu zachłannego dla różnych wartości pa- rametru $p$ . . . . .	23
7.2.1	Sposób przeprowadzenia testów . . . . .	23
7.2.2	Wyniki testów . . . . .	24
	<b>Spis rysunków</b>	<b>25</b>

# 1. Wizja produktu

## 1.1. Opis problemu

Projekt dotyczy obliczania wyników wyborów. Wybory i sposób wyłaniania zwycięzców są ściśle określone. Wybory można opisać jako parę  $(C, V)$ , gdzie  $C = \{c_1, c_2, \dots, c_m\}$  stanowi zbiór kandydatów, a  $V = (v_1, v_2, \dots, v_n)$  to ciąg wyborców. Każdy z wyborców posiada swoje preferencje wyborcze, które są opisane przez ciąg kandydatów ze zbioru  $C$ . Długość ciągu kandydatów wynosi  $m$ , a kandydaci są uporządkowani od najbardziej preferowanego do najmniej preferowanego. Ponadto określony jest rozmiar  $k$  zwycięskiego komitetu, który określa liczbę kandydatów, którzy zwyciężyli w wyborach.

### Przykładowe wybory

$$C = \{a, b, c, d, e, \dots\}$$

$$V = (v_1, v_2, v_3, \dots, v_n)$$

$$v_1 : a > b > c > d > e > \dots$$

$$v_2 : c > d > a > b > f > \dots$$

$$v_3 : \dots$$

$$\vdots$$

$$v_n : f > g > e > b > a > \dots$$

$$k = 20$$

Ciąg kandydatów przyporządkowany danemu wyborcy traktowany jest jako głos w wyborach. Każdemu kandydatowi w ciągu stanowiącym głos, przyporządkowane są punkty według punktacji Bordy. Jeżeli przez  $i$  zostanie oznaczona pozycja danego kandydata w ciągu stanowiącym głos (jeżeli kandydat jest pierwszy w ciągu, wtedy  $i = 1$ , jeżeli drugi, wtedy  $i = 2$  itd.), to wartość punktowa przyporządkowana według metody Bordy w tym głosie dla tego kandydata wynosi  $\beta(i) = m - i$ , gdzie  $m$  to rozmiar zbioru  $C$  (liczba wszystkich kandydatów). Zatem dla danego głosu kolejni uporządkowani kandydaci otrzymują kolejno  $m - 1, m - 2, \dots, 1, 0$  punktów.

### Preferencje wyborcy

$$v_1 : \overset{m-1}{a} > \overset{m-2}{b} > \overset{m-3}{c} > \overset{m-4}{d} > \dots$$

W celu wyłonienia zwycięzców wyborów komitetów definiuje się tzw. funkcję satysfakcji, która bazuje między innymi na punktacji metodą Bordy. Funkcja satysfakcji określa zadowolenie danego wyborcy ze zwycięskiego komitetu. W celu zwięzłego zdefiniowania funkcji celu wprowadza się pojęcie ciągu pozycji, które określa dla danego wyborcy pozycje wszystkich zwycięskich kandydatów z jego preferencji. Ciąg pozycji jest posortowany rosnąco - pozycja najbardziej preferowanego kandydata spośród zwycięzców na początku ciągu, pozycja najmniej preferowanego kandydata spośród zwycięzców na końcu ciągu.

### Przykład

Dla następujących preferencji:

#### Preferencje wyborcy

$$v_3 : \overset{m-1}{g} > \overset{m-2}{d} > \overset{m-3}{e} > \overset{m-4}{b} > \overset{m-5}{f} > \overset{m-6}{a} > \dots$$

komitetu  $K = \{a, b, e\}$  i wyborcy  $v_3$  ciąg pozycji oznaczany  $pos_{v_3}$  wynosi  $(3, 4, 6)$ , gdyż pozycja najlepszego kandydata ze zbioru  $K$  wynosi 3 (kandydat  $e$ ), druga najlepsza to pozycja nr 4 (kandydat  $b$ ), a ostatnią pozycją najmniej preferowanego kandydata jest pozycja nr 6 (kandydat  $a$ ).

W tym momencie można zdefiniować funkcję satysfakcji, która jako argument przyjmuje zdefiniowany przed chwilą ciąg pozycji, który dla każdego wyborcy i dla każdego komitetu jest określony jednoznacznie. Funkcja satysfakcji określająca zadowolenie danego wyborcy ze zwycięskiego komitetu, będzie więc oznaczana jako  $f(i_1, i_2, \dots, i_k)$ .

Poprzez różne zdefiniowanie funkcji satysfakcji, definiowane są odmienne systemy wyborcze. Znanymi systemami wyborczymi są system  $k$  – Borda oraz system Chamberlina-Couranta. Funkcje satysfakcji dla tych systemów określone są odpowiednio:

$$f_{k-Borda}(i_1, i_2, \dots, i_k) = \beta(i_1) + \beta(i_2) + \dots + \beta(i_k) \quad (1)$$

$$f_{CC}(i_1, i_2, \dots, i_k) = \beta(i_1) = \max(\beta(i_1), \beta(i_2), \dots, \beta(i_k)) \quad (2)$$

Wyniki wyborów dla podanych systemów mogą być różne (i zazwyczaj są) pomimo takich samych danych wejściowych. Inaczej rzecz ujmując, zwycięskie komitety mogą składać się z różnych kandydatów po zastosowaniu odmiennych funkcji satysfakcji.

Funkcja satysfakcji, która dotyczy niniejszego projektu inżynierskiego jest oparta na normie  $\ell_p$ . Norma  $\ell_p$  zdefiniowana jest następująco:

$$\ell_p(x_1, x_2, \dots, x_n) = \sqrt[p]{x_1^p + x_2^p + \dots + x_n^p}, \quad (3)$$

gdzie  $x_1, x_2, \dots, x_n \in \mathbb{R}, p \in \mathbb{N}$

Dla skrajnych przypadków, gdy  $p = 1$  i gdy  $p \rightarrow \infty$  norma  $\ell_p$  przyjmuje odpowiednio postać sumy:

$$l_1(x_1, x_2, \dots, x_n) = x_1 + x_2 + \dots + x_n \quad (4)$$

oraz postać operatora maksimum:

$$l_\infty(x_1, x_2, \dots, x_n) = \max(x_1, x_2, \dots, x_n) \quad (5)$$

Mając zdefiniowaną normę  $\ell_p$  można określić funkcję satysfakcji będącą tematem pracy.

$$f_{\ell_p-Borda}(i_1, i_2, \dots, i_k) = \ell_p(\beta(i_1), \beta(i_2), \dots, \beta(i_k)) = \sqrt[p]{[\beta(i_1)]^p + [\beta(i_2)]^p + \dots + [\beta(i_k)]^p} \quad (6)$$

Funkcja jest uogólnieniem systemu  $k - Borda$ . Jeżeli za  $p$  przyjęta zostanie wartość 1, zdefiniowana funkcja przyjmuje postać funkcji satysfakcji dla zwykłego systemu  $k - Borda$ . Dla  $p$  dążącego do nieskończoności funkcja przyjmuje postać funkcji satysfakcji dla systemu *Chamberlina-Couranta*.

## 1.2. Możliwości

Obliczanie wyników wyborów dla systemu  $\ell_p - Borda$  metodą typu brute-force jest czasochłonne. Już dla stosunkowo niewielkich rozmiarów danych, algorytm jest nieużyteczny. Dlatego głównym zadaniem projektu jest zaprojektowanie i implementacja algorytmów heurystycznych, które możliwie dokładnie i możliwie szybko obliczają wyniki wyborów. Ponadto do zadań zespołu należy próba oceny działania stworzonych algorytmów heurystycznych.

## 1.3. Motywacja do stworzenia projektu

Jednym z głównym celów realizacji projektu jest chęć poznania różnych sposobów tworzenia algorytmów heurystycznych oraz zastosowanie ich do rozwiązania problemów w interesującej zespół tematyce, jaką jest sposób wyłaniania zwycięzców w wyborach i związek między tym sposobem a satysfakcją wyborców ze zwycięzców. Ponadto zespół chciał porównać pod różnymi względami różne algorytmy heurystyczne. Interesujące jest zestawienie algorytmów pod względem dokładności obliczanych rozwiązań, czasu działania, czy trudności w projektowaniu i implementacji.

Drugą, istotną motywacją do stworzenia systemu jest możliwość wykorzystania go do wielu bardzo różnych sytuacji. Zdefiniowanie wyborów, których wyniki ma liczyć system, może dotyczyć różnego rodzaju wyborów. Mogą to być wybory ludzi do wszelakiego typu organów władz na różnych szczeblach organizacji państwowych lub organizacji prywatnych. Wybory nie muszą dotyczyć ludzi, a mogą polegać na selekcji innych rzeczy. Można zdefiniować wybory na filmy, które zostaną odtworzone w trakcie podróży samolotem, czy podczas seansu z rodziną i znajomymi. Mogą to być wybory na miejsca wspólnego wypadu znajomych na weekend. Wszędzie, gdzie ma sens zastosowanie preferencji (ułożenie rzeczy od najbardziej do najmniej pożądaney) można wykorzystać stworzony system.

Dzięki stworzonemu systemowi według określonych wymagań, możliwe jest nie tylko szybkie obliczenie wyborów na bazie wprowadzonych do systemu preferencji. Możliwe jest również sterowanie sposobem obliczania wyborów według pożądaney reprezentatywności zwycięskich kandydatów. Można decydować czy zwycięzcy wyborów będą tacy, że każdy znajdzie wśród nich takiego, którego bardzo lubi, ale też taki, którego bardzo nie lubi. Czy może zwycięzcy mają być tacy, że większość wyborców co prawda nie znajdzie wśród nich bardzo pożądanego kandydata, ale jednocześnie nie znajdzie takiego, którego bardzo nie lubi? Możliwy jest również wybór pośredni między wskazanymi dwoma rozwiązaniami.

Przykładowo, jeżeli użytkownik chce wybrać kilka filmów do wspólnego seansu z przyjaciółmi i rodziną, to może zdecydować między różnymi opcjami. Jedną z opcji jest taki wybór, aby wśród wybranych filmów każdy ze znajomych znalazł film, który mu bardzo odpowiada, lecz jednocześnie znalazły się takie, które większości znajomych wyraźnie nie odpowiadają. Druga możliwość to taka, w której wybrane zostaną filmy, które większości znajomych ani za bardzo nie odpowiadają, ani za bardzo nie przeszkadzają. Pozostałymi możliwościami są opcje pośrednie między wskazanymi.

## 1.4. Użytkownicy

Potencjalnymi użytkownikami systemu mogą być osoby i organizacje, które potrzebują systemu wybierania kilku elementów spośród puli wszystkich elementów w taki sposób, aby usatysfakcjonować swoich klientów. System może mieć przykładowe zastosowanie dla przedsiębiorstw realizujących usługi transportu lotniczego czy autokarowego do umilenia czasu podróży poprzez wybór odpowiednich filmów czy potraw. Ponadto system może służyć pojedynczym osobom w celu wyboru odpowiedniej formy spędzania wolnego czasu w grupie znajomych.

## 1.5. Usługi dostarczanie przez system

### 1.5.1. Usługi wymagane

- możliwość wprowadzenia wyborów do systemu
- obliczenie wyników wyborów
- wyświetlanie wyników wyborów

### 1.5.2. Usługi pożądane

- możliwość zarządzania swoimi wyborami (dodawanie nowych wyborów, nowych wyników, porównywanie różnych rezultatów)
- intuicyjna nawigacja po systemie
- wygodny sposób definiowania wyborów - zadanie projektowe koncentruje się na projektowaniu i tworzeniu algorytmów obliczających wyniki wyborów. System musi wczytywać pewien format pliku wejściowego definiującego wybory, których wyniki niekoniecznie są klarowne do porównania dla różnych wartości parametru  $p$ . Generowanie wyborów, których wyniki w łatwy i klarowny sposób mogłyby być wizualizowane i porównywane ułatwiłoby ocenę działania systemu oraz uatrakcyjniło korzystanie z niego

### 1.5.3. Usługi dodatkowe

- wygodny sposób porównywania wyników wyborów

## 2. Studium wykonalności

### 2.1. Opis wymagań

#### 2.1.1. Wymagania funkcjonalne

Poniżej zamieszczona jest lista najważniejszych wymagań, które wyewoluowały do przedstawionej postaci w trakcie trwania projektu. Wymagania są przedstawione w formie tabel.

Tabela składa się z trzech pól opisujących dane wymaganie: *Nazwa*, *Opis* i *Uzasadnienie*. Pole *Nazwa* jest nazwą usługi dostarczanej przez system, pole *Opis* szczegółowo opisuje w jaki sposób jest realizowane dane wymaganie, a pole *Uzasadnienie* podaje powód dlaczego w taki, a nie inny sposób wymaganie zostało zrealizowane i postawione.



Nazwa	Zdefiniowanie wyborów
Opis	<p>System oferuje różne sposoby zdefiniowania i wprowadzenia wyborów do systemu. Pierwszym i podstawowym jest możliwość określania wyborów w pliku formatu <i>.soc</i>. Sposób określania w nim kandydatów oraz głosów wyborców jest ściśle określony. W pierwszej linii pliku znajduje się liczba kandydatów do zwycięskiego komitetu. W kolejnych liniach znajdują się numery i nazwy kandydatów. Następnie jest linia z informacją o liczbie głosujących, liczbie policzonych głosów oraz liczbie głosów unikalnych. W ostatnim bloku pliku znajdują się linie z informacją o preferencjach głosów unikalnych. Każda linia składa się z liczby powtórzeń danego głosu oraz preferencji określonej dla tego głosu, która ma postać ciągu numerów kandydatów od najbardziej do najmniej pożądanego. Po stworzeniu opisanego pliku system umożliwia wskazanie pliku z systemu plików.</p> <p>Drugim sposobem definiowania wyborów oferowanym przez system jest generowanie wyborów z rozkładu normalnego. W tym celu kandydaci i wyborcy są reprezentowani jako punkty na płaszczyźnie. Preferencje wyborców obliczane są na bazie odległości euklidesowych do poszczególnych kandydatów na płaszczyźnie. Aby wygenerować wybory użytkownik podaje w formularzu parametry potrzebne do generacji wyborów zgodnie z rozkładem normalnym. Uzupełnia pola dotyczące: liczby kandydatów, średniej wartości współrzędnej <math>x</math> dla wyborców, średniej wartości współrzędnej <math>x</math> dla kandydatów, średniej wartości współrzędnej <math>y</math> dla wyborców, średniej wartości współrzędnej <math>y</math> dla kandydatów, odchylenia standardowego wyborców oraz odchylenia standardowego kandydatów.</p>
Uzasadnienie	Definicja wyborów za pomocą pliku formatu <i>.soc</i> zapewnia spełnienie podstawowej usługi oferowanej przez system - wprowadzenia wyborów do systemu. Uzasadnieniem dla konieczności generacji wyborów z rozkładu normalnego jest możliwość atrakcyjnej i klarownej wizualizacji wyborów i wyników wyborów. Wizualizacja w ten sposób wyników wyborów umożliwia między innymi ocenienie jakości stworzonych algorytmów i porównanie ich.

Tabela 1: Zdefiniowanie wyborów

Nazwa	Obliczanie wyników wyborów
Opis	System oferuje kilka algorytmów heurystycznych obliczających wyniki wyborów. Algorytmy bazują na różnym paradygmacie tworzenia ich. Jeden z paradygmatów to programowanie zachłanne, a drugi to programowanie genetyczne. Użytkownik przy tworzeniu nowego wyniku dla zdefiniowanych wyborów może w prosty sposób wybrać jeden z algorytmów.
Uzasadnienie	Kilka stworzonych algorytmów i różne podejście do tworzenia ich, umożliwia ocenę jakości algorytmów na bazie zestawienia wyników obliczonych wyborów. Ponadto stworzenie różnych algorytmów pozwoli na dogłębsze poznanie sposobów tworzenia algorytmów heurystycznych, jak również samej tematyki pracy, jaką były różne sposoby wyboru zwycięskiego komitetu.

Tabela 2: Obliczanie wyników wyborów

<b>Nazwa</b>	Wersja algorytmu zachłannego <i>Greedy</i> - <i>CC</i>
<b>Opis</b>	Implementacja algorytmu zachłannego dla szczególnego przypadku (parametr $p \rightarrow \infty$ )
<b>Uzasadnienie</b>	Dla stosunkowo “dużych” wartości parametru $p$ wykonywanie operacji pierwiastkowania i potęgowania stanowi niepotrzebny narzut czasowy. W tym przypadku operację obliczania normy $\ell_p$ można zastąpić wyznaczaniem maksimum dla pojedynczych wyników kandydatów z komitetu, dla którego liczono by normę $\ell_p$ . Korzyści jakie płyną z takiej modyfikacji to nie tylko skrócenie czasu obliczeń, ale także ustalanie od jakiej wartości parametru $p$ komitet wygrywający jest bliski przybliżonego wyniku otrzymanego dla systemu <i>Chamberlina-Couranta</i> .

Tabela 3: Wersja algorytmu zachłannego *Greedy* - *CC*

<b>Nazwa</b>	Prezentacja wyników wyborów
<b>Opis</b>	<p>System oferuje różne sposoby prezentacji wyników wyborów. Pierwszy z nich, to proste wskazanie kandydatów i wypisanie nazw kandydatów w widocznym panelu. Ta forma prezentacji dotyczy wyników wyborów, które zdefiniowane zostały za pomocą pliku formatu <i>.soc</i>.</p> <p>Druga forma prezentacji wyników wyborów dotyczy wyborów wygenerowanych z rozkładu normalnego. Wizualizacja polega na narysowaniu wykresu <math>2D</math>, na którym punkty reprezentujące kandydatów ze zwycięskiego komitetu są wyraźnie wyróżnione na tle kandydatów, którzy nie dostali się do zwycięskiego komitetu. Dla wyborów generowanych z rozkładu normalnego system również oferuje wypisanie nazw zwycięzców w dobrze widocznym panelu.</p> <p>Ponadto system oferuje tabelę z wszystkimi wynikami wyborów obliczonych przez różne algorytmy i dla różnych parametrów <math>p</math>. Wyniki dla różnych parametrów <math>p</math> są wyraźnie oddzielone.</p>
<b>Uzasadnienie</b>	<p>Prezentacja wyników wyborów w postaci wykresu <math>2D</math> z wyraźnie zaznaczonymi zwycięzcami i przegranymi, którzy są reprezentowani jako punkty, pozwala na porównanie w prosty sposób wyników wyborów dla różnych wartości parametru <math>p</math>. Taka forma wizualizacji jest jednocześnie wizualnie atrakcyjna. Prezentacja wyników dla wyborów zdefiniowanych z pliku formatu <i>.soc</i> ogranicza się jedynie do wypisania zwycięzców, ponieważ kandydaci nie są reprezentowani jako punkty i tym samym nie mają współrzędnych w tej odmianie wyborów.</p> <p>Prezentacja wszystkich wyników w postaci tabeli z wyraźnie oddzielnymi wynikami dla różnych wartości parametru <math>p</math>, również ułatwia porównywanie wyników między różnymi algorytmami.</p>

Tabela 4: Prezentacja wyników wyborów

### 2.1.2. Wymagania niefunkcjonalne

Wymagania niefunkcjonalne podzielono na wymagania produktowe i wymagania organizacyjne. Poniżej zamieszczono tabele analogiczne do tych z wymagań funkcjonalnych dla odpowiednich typów wymagań niefunkcjonalnych.

#### Wymagania produktowe

<b>Nazwa</b>	Obliczanie wyników wyborów w satysfakcjonującym czasie
<b>Opis</b>	Ponieważ dla już stosunkowo małych danych metoda brute-force rozwiązywania opisanego problemu nie daje rezultatów w satysfakcjonującym czasie, więc to wymaganie niefunkcjonalne jest kluczowe w tym projekcie. Zastosowanie algorytmów heurystycznych wychodzi naprzeciw temu wymaganiu. Algorytmy powinny dawać rozwiązania w zadowalającym czasie dla możliwie dużych danych.
<b>Uzasadnienie</b>	Czas działania algorytmu powinien się mieścić w możliwym do zaakceptowania przedziale czasu ze względu na konieczność jego użyteczności.

Tabela 5: Obliczanie wyników wyborów w satysfakcjonującym czasie

#### Wymagania organizacyjne

<b>Nazwa</b>	Dotrzymanie terminów na przedstawienie poszczególnych elementów pracy
<b>Opis</b>	Kolejne elementy pracy, które należało przedstawiać w umówionych terminach to: wizja produktu, studium wykonalności, prototyp oraz tuż przed końcem projektu: produkt końcowy wraz z dokumentacją.
<b>Uzasadnienie</b>	Terminy są dostosowane do trybu wykonywania pracy inżynierskiej, który jest ustalany przez władze uczelni.

Tabela 6: Obliczanie wyników wyborów w satysfakcjonującym czasie

## 2.2. Strategia testowania

Podstawowym rodzajem testów pisanych w trakcie tworzenia systemu były testy jednostkowe. Były one dodawane na bieżąco od razu przy dodawaniu kolejnych funkcjonalności systemu.

Ponadto w celu zagwarantowania poprawności działania wcześniej dodanych funkcjonalności po dodaniu nowych skorzystano z ciągłej integracji. Do tego celu wykorzystano serwis *Travis CI* skonfigurowany do repozytorium kodu. Przy każdej zmianie kodu źródłowego w repozytorium, dokonywane było automatyczne włączenie testów jednostkowych.

W celu oceny jakości obliczanych wyników wyborów przez stworzone algorytmy, przeprowadzono testy porównawcze. Zestawiano wyniki wyborów obliczonych przez różne algorytmy dla tych samych danych wejściowych. Ponadto porównywano wykresy wyników wyborów z oczekiwanymi rezultatami.

### 2.3. Aspekt technologiczny

Do stworzenia systemu wykorzystano język *Python 2.7*. Produkt postanowiono wykonać w postaci aplikacji webowej. Aplikację internetową oparto na frameworku pythonowym *Django 1.9*. Do tworzenia stron internetowych użyto frameworku *Bootstrap*. Do wdrożenia systemu wykorzystano platformę *Heroku*.

Wybrano taki stos technologiczny ze względu na doświadczenie części zespołu pracy z nim. Wiedza na temat tych narzędzi przekonywała zespół o możliwości zrealizowania projektu w tej technologii.

### 2.4. Analiza ryzyka

Głównym zagrożeniem dla realizacji produktu mogły być problemy ze stworzeniem wystarczająco dobrego algorytmu heurystycznego dla opisanego problemu. Ze względu na niewielkie doświadczenie zespołu w tej dziedzinie, opisana sytuacja z dużym prawdopodobieństwem mogła wystąpić. Głównym zabezpieczeniem na wypadek takiego scenariusza było stosunkowo szybkie stworzenie bazy całego systemu i możliwość wczesnego skupienia się na algorytmach heurystycznych.

Innym, dużo mniejszym zagrożeniem dla projektu mógł być brak skoordynowania prac poszczególnych członków zespołu. Ponieważ dla członków zespołu projekt inżynierski był tylko jednym z wielu zajęć w czasie trwania projektu, mogła wystąpić sytuacja kiedy ze względu na natężenie innych obowiązków, dany członek zespołu nie mógłby wykonać koniecznej pracy w odpowiednim czasie.

## 3. Wybrane aspekty realizacji

### 3.1. Lista modułów

Moduły przedstawione są w postaci przyporządkowania do modułów usług, za które w systemie odpowiedzialny jest dany moduł. Część opisanych czynności jest wykonywana poprzez współpracę różnych modułów.

#### 1. Moduł administracji kont

- logowanie
- wylogowanie
- rejestracja

#### 2. Moduł zarządzania wyborami

- wyświetlenie listy wszystkich wyborów
- wyświetlenie listy wszystkich wyników dla danych wyborów
- stworzenie nowych wyborów
- stworzenie nowego wyniku
- usunięcie wyborów
- usunięcie wyniku

#### 3. Moduł wizualizacji wyborów i wyników wyborów

- wizualizacja wyborów wygenerowanych zgodnie z rozkładem normalnym
- wizualizacja wyników wyborów dla wyborów wygenerowanych zgodnie z rozkładem normalnym oraz zaimportowanych z pliku w formacie *.soc*

#### 4. Moduł generacji i wczytywania wyborów

- generacja wyborów z rozkładu normalnego
- wczytywanie wyborów z pliku formatu *.soc*

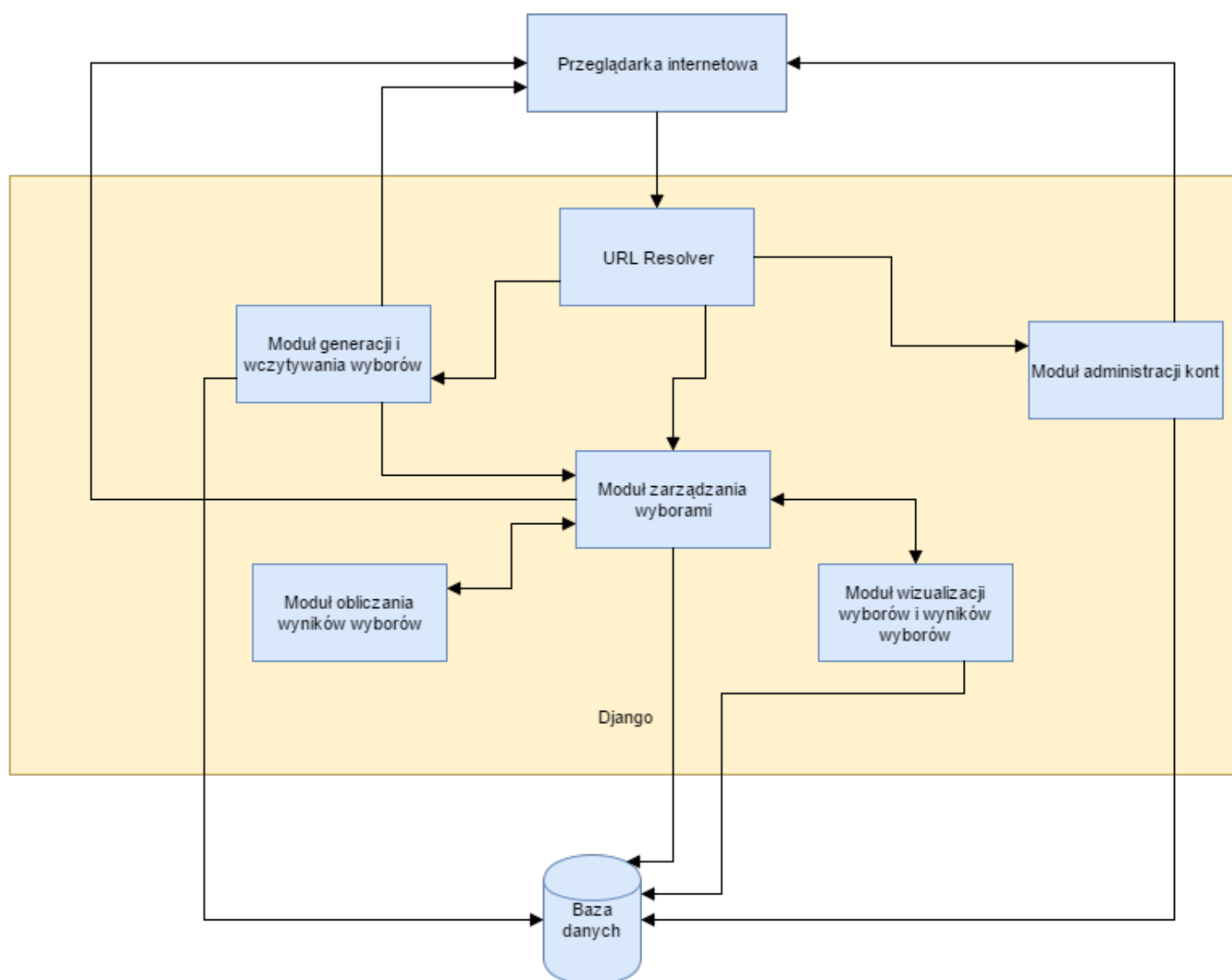
#### 5. Moduł obliczania wyników wyborów

- obliczanie wyników wyborów za pomocą różnych algorytmów

#### 6. Moduł URL Resolver

- przekazywanie żądań otrzymywanych przez klienta do odpowiednich modułów

### 3.2. Diagram komunikacji



Rysunek 1: Diagram komunikacji

### 3.2.1. Opis diagramu

Na diagramie oprócz przytoczonych modułów znajdują się przeglądarka internetowa oraz baza danych. W celu wykonania swoich usług moduły współpracują z tymi dodatkowymi komponentami diagramu. Wszystkie moduły wykorzystują wewnętrzne mechanizmy *Django*, dlatego znajdują się wewnątrz dużego obszaru oznaczonego jako *Django*. Największym modułem spośród wszystkich modułów jest moduł zarządzania wyborami. Współpracuje on z innymi modułami w celu wykonania niektórych swoich zadań. Zleca on między innymi obliczenie wyników wyborów modułowi obliczania wyników wyborów przy wykonywaniu zadania stworzenia nowego wyniku. Najbardziej odosobnionym modułem jest moduł administracji kont, który nie współpracuje z innymi modułami. Jedynie moduł URL Resolver kieruje odpowiednie zapytania do tego modułu i jest to jedyna komunikacja z modułem administracji kont. Większość modułów w celu realizacji swoich zadań komunikuje się z bazą danych. Dokładny opis diagramu ze ścieżkami przejść znajduje się w dokumentacji technicznej.

## 4. Organizacja pracy

### 4.1. Metodyka pracy

Tworzenie systemu będącego realizacją tego projektu inżynierskiego można uznać za proces ewolucyjnego tworzenia oprogramowania. Klient na początku projektu określił wymagania w bardzo ogólny sposób. Tworzony system miał za zadanie obliczać wyniki wyborów dla zdefiniowanego przez klienta systemu wyborczego za pomocą algorytmów heurystycznych. W miarę postępu projektu, wymagania były precyzowane na bazie stałego kontaktu z klientem i potrzeb rozwijanego systemu. Powstawały kolejne wersje systemu aż do osiągnięcia systemu końcowego.

### 4.2. Podział prac

W ramach realizacji projektu przydzielano prace do poszczególnych członków zespołu.

Tomasz Kasprzyk

- generacja wyborów z rozkładu normalnego
- wizualizacja wyników wyborów i wydajności algorytmów na wykresach
- współtworzenie dokumentacji projektowej
- tworzenie testów jednostkowych i porównawczych
- współtworzenie prezentacji na zajęcia seminaryjne

Daniel Ogiela

- projekt i implementacja algorytmu zachłannego zależnego od parametru  $p$
- implementacja algorytmu zachłannego według zasady *Chamberlina-Couranta*
- współtworzenie dokumentacji projektowej
- tworzenie testów jednostkowych i porównawczych
- współtworzenie prezentacji na zajęcia seminaryjne

Jakub Stępak

- stworzenie szkieletu aplikacji
- konfiguracja ciągłej integracji
- implementacja algorytmu *brute-force*
- implementacja algorytmu genetycznego
- przedstawienie wyników na wykresie i tabelaryczne
- współtworzenie testów jednostkowych i porównawczych
- współtworzenie dokumentacji projektowej
- współtworzenie prezentacji na zajęcia seminaryjne

### 4.3. Narzędzia do zarządzania projektem

Do przydzielania zadań członkom zespołu i ustalania ich terminów wykorzystano aplikację internetową *Trello*. Do przechowywania kodu źródłowego oprogramowania wykorzystano serwis internetowy *GitHub*. W celu zdalnego kontaktowania się pomiędzy członkami zespołu korzystano z oprogramowania *Slack*. Korzystano z aplikacji *Google Docs* do tworzenia roboczej wersji dokumentacji. Właściwa wersja dokumentacji została stworzona w *LaTeX-u* i do jej przygotowania wykorzystano program *TexMaker*. Do przygotowywania diagramów do dokumentacji wykorzystano aplikację *draw.io*.

### 4.4. Opis kolejnych wersji systemu

#### 4.4.1. Wersja 1.

Pierwsza wersja systemu była uruchamiana z poziomu konsoli *Pythona*. Aplikacja przyjmowała za argumenty parametr  $p$  konieczny do obliczania normy  $\ell_p$  oraz plik z rozszerzeniem *.soc* definiujący wybory. Aplikacja potrafiła obliczyć wyniki wyborów dla małych danych za pomocą algorytmu typu *brute-force*. Pierwsza wersja systemu nie spełniała wymagań stawianych systemowi. Powstała w celu dookreślenia wymagań.

#### 4.4.2. Wersja 2.

Druga wersja systemu to aplikacja webowa skonfigurowana na platformie *Heroku*. Aplikacja posiadała funkcjonalności poprzedniej wersji systemu oraz miała dodany szereg nowych funkcjonalności. Pozwalała na zarządzanie kontami użytkowników. Każdy użytkownik mógł mieć swoje konto i mógł zarządzać swoimi wyborami. Miał możliwość tworzenia wyborów, usuwania wyborów, dodawania nowych wyników wyborów czy wyświetlania zwycięzców. Ponadto użytkownik mógł generować wybory zgodnie z rozkładem normalnym według podanych parametrów. Dla tak stworzonych wyborów użytkownik miał możliwość wyświetlić wykresy  $2D$ , na których kandydaci i wyborcy byli reprezentowani za pomocą punktów. Ta wersja stanowiła bazę dla kolejnych wersji systemu.

#### 4.4.3. Wersja 3.

Trzecia wersja systemu dodała do funkcjonalności pierwsze wersje algorytmów heurystycznych. Użytkownik mógł obliczać wyniki wyborów za pomocą algorytmu zachłannego i genetycznego. System pozwalał na uzyskanie wyników wyborów dla większych danych i dawał wizualnie zadowalające rezultaty.

#### **4.4.4. Wersja 4 - końcowa.**

Końcowa wersja systemu usprawniła wcześniejsze algorytmy heurystyczne oraz dodała do funkcjonalności jeszcze jeden algorytm heurystyczny, który skutecznie i szybko obliczał wyniki wyborów dla odpowiednio dużego parametru  $p$ . Dodano ponadto liczne funkcjonalności poprawiające łatwość i intuicyjność korzystania z aplikacji. System pozwalał na przeglądanie wyników danych wyborów za pomocą suwaka oraz na usuwanie pojedynczych wyników. Do każdego elementu listy rezultatów dodano informację o wyniku punktowym. Ponadto poprawiono wygląd strony.



## 5. Opisy algorytmów heurystycznych

### 5.1. Algorytm zachłanny

Algorytm aproksymuje wyniki wyborów na bazie funkcji satysfakcji systemu  $\ell_p - Borda$ . Działa w ten sposób, że główna pętla algorytmu w każdej iteracji wybiera jednego kandydata do zwycięskiego komitetu. Jeżeli dany kandydat zostanie wybrany w którejś pętli do zwycięskiego komitetu, na pewno znajdzie się wśród ostatecznych zwycięzców. Kolejni kandydaci dobierani są w ten sposób do komitetu, że badane jest ile poszczególni, jeszcze niewybrani kandydaci wnoszą zadowolenia wyborców do wcześniej wybranych zwycięzców. Ten z kandydatów, który wnosi najwięcej zadowolenia wyborców wygrywa daną iterację i tym samym wchodzi do zwycięskiego komitetu.

To ile wnosi dany kandydat do ogólnego zadowolenia wyborców ustalane jest na bazie wspomnianej na początku funkcji satysfakcji. Wykorzystuje się ją do ustalenia pojedynczego zadowolenia danego wyborcy z komitetu złożonego z wybranych we wcześniejszych iteracjach kandydatów oraz badanego kandydata. Tak otrzymywane pojedyncze zadowolenia dla wszystkich wyborców sumuje się. Kandydat, którego suma będzie w danej iteracji największa, wygrywa tą iterację i wchodzi do zwycięskiego komitetu. Główna pętla algorytmu trwa aż cały komitet zostanie uzupełniony.

### 5.2. Algorytm zachłanny według zasady Chamberlina-Couranta

Na początku należy zaznaczyć, że ten algorytm nie aproksymuje wyników wyborów na bazie funkcji satysfakcji systemu  $\ell_p - Borda$ , która jest głównym tematem projektu. Wyniki są aproksymowane za pomocą funkcji satysfakcji *Chamberlina-Couranta* (szczegółowy opis w rozdziale *Opis problemu*), która jest szczególnym przypadkiem funkcji satysfakcji systemu  $\ell_p - Borda$  dla  $p \rightarrow \infty$ . W związku z tym na działanie tego algorytmu nie wpływa parametr  $p$ , który jest jedną z danych wejściowych do obliczania wyników wyborów. Jest to najważniejsza różnica między tym, a powyżej opisanym algorytmem, który różnicuje swoje działanie w zależności od parametru  $p$ .

Schemat działania tego algorytmu jest właściwie identyczny jak opisanego powyżej. W algorytmie występuje główna pętla, która w każdej iteracji wybiera jednego kandydata. W danej iteracji dobierany jest najlepszy kandydat pod względem dodanego dodatkowego ogólnego zadowolenia wyborców. Dodatkowe zadowolenie jest obliczane za pomocą funkcji satysfakcji *Chamberlina-Couranta*. Dość istotna różnica między tym a powyżej opisanym algorytmem jest taka, że pojedyncze zadowolenie wyborcy na danym etapie algorytmu może być zwiększone tylko wtedy, gdy badany kandydat jest wyżej w preferencji wyborcy od każdego z wcześniej wybranych kandydatów. Ta różnica wynika z wykorzystania innej funkcji satysfakcji.

### 5.3. Algorytm genetyczny

Ideą algorytmu genetycznego jest poddawanie „osobników” - w tym wypadku wybranych komitetów (podzbiorów kandydatów) - działaniom analogicznym do ewolucji w naturze. Osobniki są więc krzyżowane (wybierani są losowo kandydaci do nowego komitetu z dwóch już istniejących) lub mutowane (zmieniany jest losowo jeden z kandydatów w komitecie). Te procesy powtarza się kilkadziesiąt - kilkaset razy, za każdym razem pozostawiając w puli tylko określoną liczbę najlepszych osobników.

Po zakończeniu działania najlepszy z pozostałych w puli kandydatów jest podawany jako wynik działania algorytmu.

## 6. Efekt końcowy

### 6.1. Oczekiwane rezultaty

Wyniki wyborów, pomimo takich samych preferencji wyborców, powinny się różnić w zależności od parametru  $p$ , który jest argumentem funkcji satysfakcji określonej dla uogólnienia systemu  $k - Borda$ . Charakterystyczne są dwa skrajne przypadki, w których parametr  $p$  przyjmuje wartość 1 oraz kiedy dąży do nieskończoności. Dla pierwszego przypadku funkcja satysfakcji przyjmuje postać:

$$f_{k-Borda}(i_1, i_2, \dots, i_k) = \beta(i_1) + \beta(i_2) + \dots + \beta(i_k) \quad (1)$$

Jest ona zgodna z regułą  $k - Borda$  i jest sumą przydzielonych punktów według punktacji Bordy na poszczególnych kandydatów ze zwycięskiego komitetu. Przy zastosowaniu tego systemu do wyznaczenia zwycięskiego komitetu, wyniki wyborów średnio satysfakcjonują wyborców pod względem reprezentatywności. Zwycięzcy kandydaci dobrani są w ten sposób, że dla większości wyborców nie są oni ani bardzo pożądanymi, ani przesadnie odrzucającymi. Umieszczeni są zazwyczaj w środku preferencji i w ten sposób dla większości wyborców nie są zadowalającym reprezentantem i nie różnią się zbyt. W modelu, w którym zarówno kandydaci, jak i wyborcy przedstawieni są jako punkty na płaszczyźnie, punkty reprezentujące zwycięzców oscylują w środku spektrum poglądów. Większość wyborców nie ma swojego wyraźnego reprezentanta, lecz jednocześnie wśród zwycięzców nie ma takiego, który byłby przez nich wyraźnie odrzucany.

Dla drugiego skrajnego przypadku funkcja satysfakcji przyjmuje postać:

$$f_{CC}(i_1, i_2, \dots, i_k) = \beta(i_1) \quad (2)$$














































Jest ona zgodna z regułą *Chamberlina-Couranta* i na jej wartość wpływają tylko punkty przydzielone według punktacji Bordy na najwyższej umieszczonego w preferencjach wyborczych kandydata spośród zwycięskiego komitetu. W przeciwieństwie do przypadku dla  $p$  wynoszącego 1, wyniki wyborów obliczane według tego systemu sprzyjają zasadzie reprezentatywności. Większość wyborców będzie miała wśród członków zwycięskiego komitetu swojego wyraźnego reprezentanta, którego umieściła wysoko w preferencjach wyborczych. Jednocześnie wśród zwycięzców będą tacy, którzy wyraźnie nie odpowiadają większości wyborców. W modelu opisanym wyżej, punkty reprezentujące zwycięzców wyborów będą rozproszone po całym spektrum poglądów. Wśród pewnej klasy wyborców, którzy zgromadzeni są blisko pewnego zestawu poglądów, znajdzie się co najmniej jeden zwycięzca, który najbardziej odpowiada ich poglądom.

Pomiędzy skrajnymi przypadkami znajdują się stany pośrednie, które w zależności od parametru  $p$  i specyfiki samych wyborów (liczba i rozmieszczenie kandydatów, liczba i rozmieszczenie wyborców) powinny być bardziej lub mniej zbliżone do stanów skrajnych dla  $p$  równego 1 i dla  $p \rightarrow \infty$ . Stosując model, w którym kandydaci i wyborcy są przedstawieni jako punkty na płaszczyźnie, w miarę wzrostu parametru  $p$  rozproszenie punktów reprezentujących zwycięzców powinno się zwiększać aż do osiągnięcia pewnej granicznej wartości parametru  $p$ . Dla wartości parametrów  $p$  większych od tej granicznej wartości, wyniki wyborów się już nie zmieniają.

## 6.2. Prezentacja i jakość wyników

### 6.2.1. Prezentacja tabeli z wszystkimi wynikami dla danych wyborów

Poniżej tabela z czasami i zadowoleniem wyborców z wyników wyborów obliczonych za pomocą różnych algorytmów. Testowane wybory wygenerowano z rozkładu normalnego. Zawierają 30 kandydatów, 50 wyborców oraz komitet zwycięski liczący 15 kandydatów. Wyniki obliczono dla różnych wartości parametru  $p$ . Tabela generowana i wyświetlana przez system.

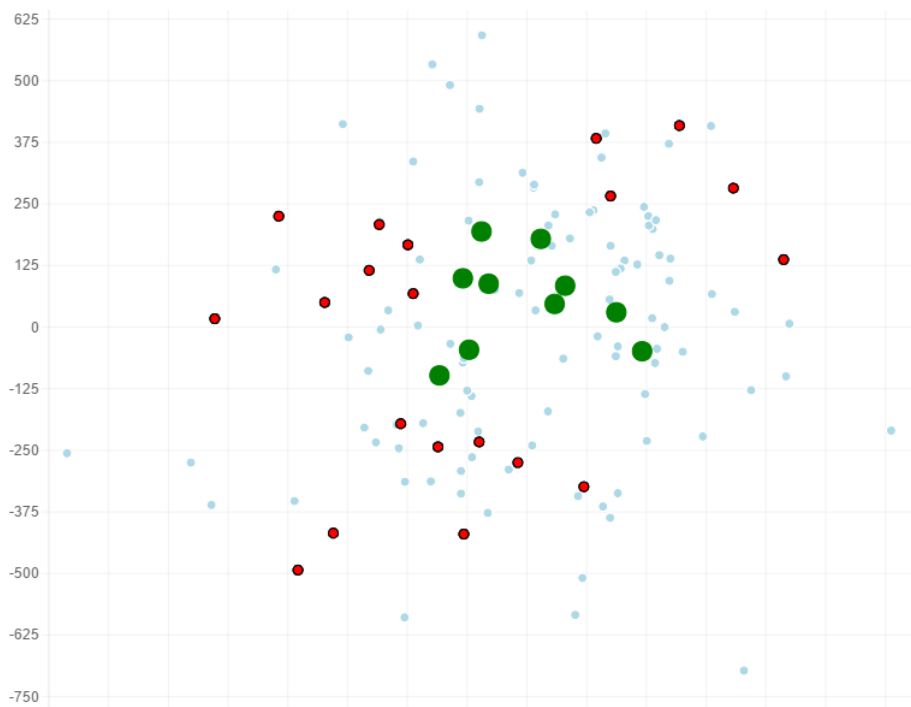
p param	Algorithm	Time	Committee score ⓘ	Algorithm parameters	Actions
1	Greedy Algorithm	2.10 s	14364.00		 
1	Greedy CC	0.13 s	12968.00		 
1	Genetic	5.24 s	14364.00	 50  10  20	 
2	Greedy Algorithm	9.09 s	3906.48		 
2	Greedy CC	0.13 s	3680.18		 
2	Genetic	15.71 s	3906.48	 50  10  20	 
5	Greedy Algorithm	10.87 s	1906.82		 
5	Greedy CC	0.18 s	1868.95		 
5	Genetic	17.84 s	1906.76	 50  10  20	 
10	Greedy Algorithm	10.84 s	1572.36		 
10	Greedy CC	0.13 s	1565.83		 
10	Genetic	18.09 s	1572.02	 50  10  20	 
15	Greedy Algorithm	9.83 s	1500.65		 
15	Greedy CC	0.13 s	1497.39		 
15	Genetic	19.29 s	1501.38	 50  10  20	 

Rysunek 2: Wyniki wyborów w postaci wyświetlonej przez system tabeli

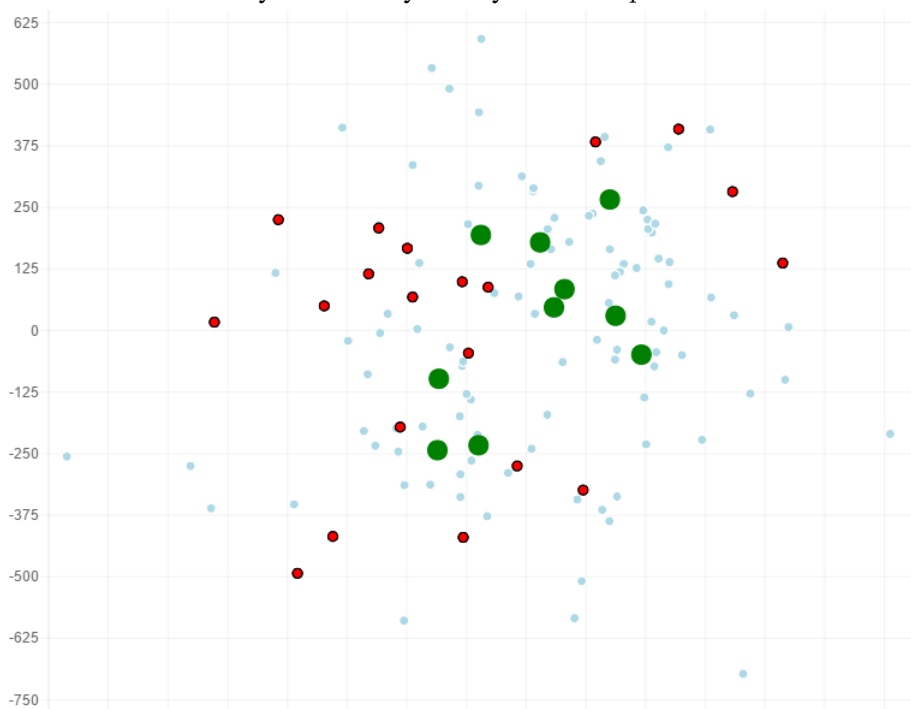
System wyświetla wyniki w taki sposób, aby wyniki wyborów wygenerowanych dla tego samego parametru  $p$  były zgrupowane razem. W tym celu wyniki wyborów dla tego samego parametru  $p$  występują obok siebie oraz mają inny kolor tła od sąsiednich wyników dla innego parametru  $p$ . Warto zwrócić uwagę, że w miarę wzrostu parametru  $p$  wartość zadowolenia wyborców z komitetów maleje. Wynika to specyfiki funkcji satysfakcji dla systemu  $\ell_p$  – Borda. Porównywanie między sobą wartości zadowoleń z danego komitetu dla różnych wartości parametru  $p$  raczej mija się z celem.

### 6.2.2. Prezentacja szczegółowych wyników wyborów

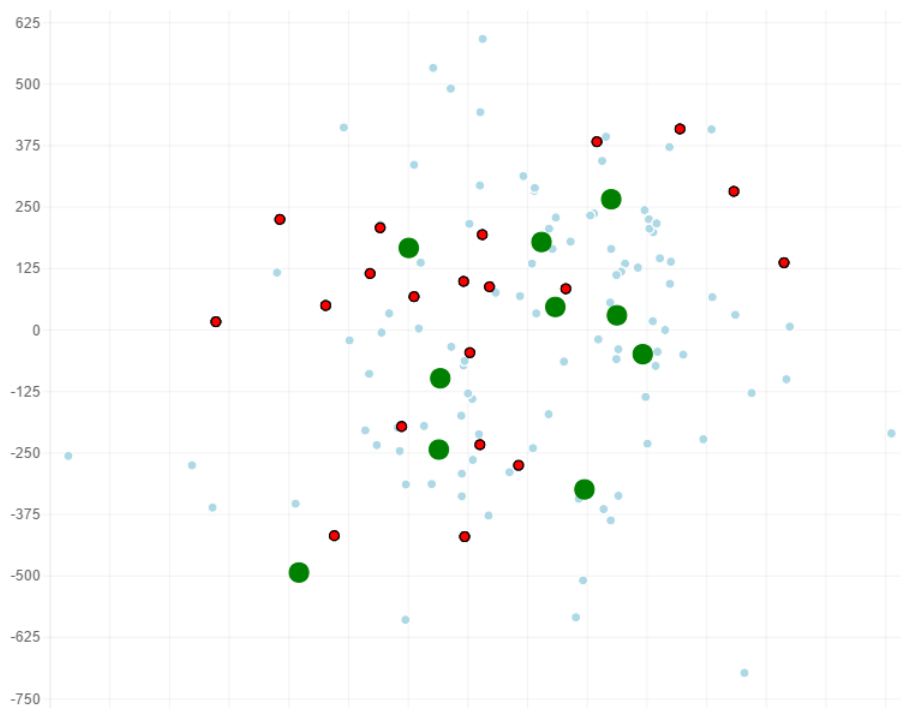
Na kolejnych grafikach przedstawiono szczegółowe wyniki wyborów w postaci wykresów 2D. Wybory wygenerowano z rozkładu normalnego. Wyborcy i kandydaci są w nich przedstawiani jako punkty na płaszczyźnie. Wybory składają się z 30 kandydatów, 100 wyborców oraz komitetu zwycięskiego liczącego 10 kandydatów. Wyświetlone wyniki obliczono za pomocą algorytmu zachłannego dla rosnących parametrów  $p$ . Grafiki pochodzą ze wcześniejszej wersji systemu.



Rysunek 3: Wyniki wyborów dla  $p = 1$



Rysunek 4: Wyniki wyborów dla  $p = 5$



Rysunek 5: Wyniki wyborów dla  $p = 20$

Na wykresach można zaobserwować stopniowe rozpraszanie się zwycięskich kandydatów dla zwiększającej się wartości parametru  $p$ .

## 7. Testy porównawcze

### 7.1. Zrównanie jakości działania algorytmów zachłannych

#### 7.1.1. Przewidywany schemat działania algorytmów zachłannych

W związku ze specyfiką zaimplementowanych algorytmów zachłannych przyjęto pewien sposób zachowania się wyników algorytmów względem siebie dla różnych wartości parametru  $p$ . Dla małych wartości parametru  $p$  algorytm zachłanny zależny od parametru  $p$  produkuje zdecydowanie lepsze jakościowo rezultaty od algorytmu według zasady *Chamberlina-Couranta*. Wraz ze wzrostem parametru  $p$  różnica w jakości działania maleje aż do osiągnięcia wartości granicznej parametru  $p$ , począwszy od której algorytm zachłanny według zasady *Chamberlina-Couranta* staje się nie gorszy od algorytmu zachłannego zależnego od parametru  $p$ . Wszystkie przypadki testowe zachowywały się zgodnie z przyjętym schematem działania. Ponieważ algorytm zachłanny według zasady *Chamberlina-Couranta* jest zdecydowanie szybszy od drugiego algorytmu zachłannego zasadna jest próba określenia wartości granicznej parametru  $p$  dla różnych parametrów wyborów.

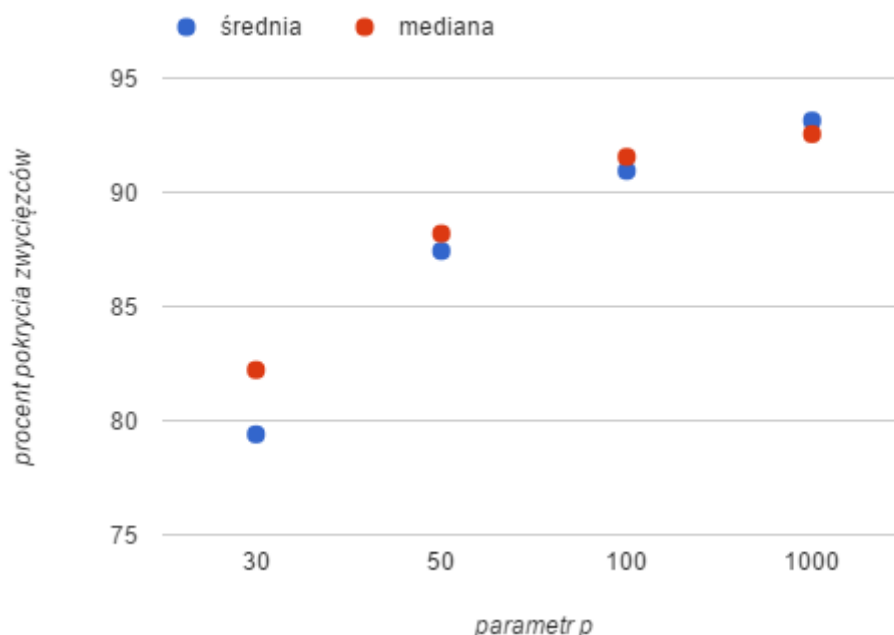
#### 7.1.2. Wartość graniczna parametru $p$

Sprawdzono zależność granicznej wartości parametru  $p$  od różnych parametrów definiujących wybory. Testowano zależność od liczby kandydatów, liczby wyborców, rozmiaru komitetu oraz odchylenia standardowego. Oddzielnie badano zależność od każdego z podanych parametrów przyjmując za pozostałe parametry pewne stałe. Po wynikach testów ciężko było wskazać zależność między wartością graniczną parametru  $p$  a którymś z badanych parametrów wyborów. Rozrzut wyników był duży i dotyczył nawet różnych prób dla tych samych danych. Wartości graniczne parametru  $p$  oscylowały od rzędu kilkadziesiąt do rzędu kilka tysięcy.

W trakcie wykonywania testów zauważono jednak inną prawidłowość. Już dla stosunkowo niewielkich wartości parametru  $p$  wyniki działania obydwu algorytmów w znacznym stopniu się pokrywały. Duża większość zwycięskich kandydatów była wspólna, a różnice dotyczyły jedynie pewnych par lub trójek kandydatów, którzy wymieniali się w roli zwycięzcy dla dużych wartości parametru  $p$ .

#### 7.1.3. Stopień pokrywania się zwycięzców dla różnych wartości parametru $p$

W celu zbadania jaki jest procentowy stopień pokrycia się komitetów, przeprowadzono testy dla różnych wartości parametru  $p$ . Dla każdej wybranej wartości parametru  $p$  przeprowadzono 10 prób. Każda próba to inaczej zdefiniowane wybory. Wśród 10 prób rozmiar komitetu znajdował się w przedziale  $[10, 55]$ , liczba kandydatów w przedziale  $[45, 150]$ , a liczba wyborców w przedziale  $[40, 150]$ . Poniżej przedstawiono na wykresie średnie i mediany pokrycia się zwycięzców. Dla każdej wartości parametru  $p$  obliczono średnią i medianę z 10 opisanych prób.



Rysunek 6: Średnie i mediany pokrycia się zwycięskich komitetów

Wartości średnich i median znajdują się blisko siebie dla każdej wartości parametru  $p$ . Rozrzut wyników dla każdej wartości parametru  $p$  nie był duży. Już dla wartości  $p = 50$  średnie pokrycie się komitetów obliczanych przez obydwa algorytmy przekroczyło 87%. Dla  $p = 100$  średnia przekroczyła 90%. Natomiast dla  $p = 1000$  średnia wzrosła nieznacznie w stosunku do średniej dla wartości  $p = 100$ . Pozostałe kilka bądź kilkanaście procent zwycięzców, którzy się nie pokrywają dla obydwu algorytmów stanowią wspomniane wcześniej pary bądź trójki kandydatów, którzy wymieniają się na pozycji zwycięzców dla dużych wartości parametru  $p$ , które są rzędu kilka tysięcy.

Wyniki wskazują, że już dla wartości  $p = 50$  można stosować algorytm według zasady *Chamberlina-Couranta* zamiast drugiego algorytmu zachłannego, mając dużą szansę na pokrycie się zwycięzców na poziomie niecałych 90%. Wartość graniczną parametru  $p$  trudno przewidzieć niezależnie od parametrów wyborów.

## 7.2. Porównanie algorytmu genetycznego i algorytmu zachłannego dla różnych wartości parametru $p$

### 7.2.1. Sposób przeprowadzenia testów

W ramach testów porównawczych sprawdzono działanie algorytmu zachłannego i algorytmu genetycznego dla różnych zestawów parametrów działania. Porównano osiem różnych kombinacji parametrów działania algorytmu genetycznego. Liczba cykli w testowanych zestawach parametrów wynosiła 20 lub 50, prawdopodobieństwo mutacji wynosiło 50% lub 100%, a zakres krzyżowania 10%, 50% lub 100%. Dla tak dobranych parametrów algorytmu genetycznego zebrano wartości wybranych komitetów i czasy działania algorytmów i porównano z wynikami i czasami algorytmu zachłannego. Wartości parametru  $p$ , dla którego przeprowadzono badania wynosiły 1, 5 i 50. Testy przeprowadzono dla wyborów zawierających 30 kandydatów, 50 wyborców oraz rozmiaru komitetu liczącego 15 zwycięzców.

### 7.2.2. Wyniki testów

Po zebraniu wyników testów, zestawiono dla każdej z badanych wartości parametru  $p$  rezultaty osiągnięte przez algorytm zachłanny oraz reprezentanta algorytmu genetycznego dla danej wartości parametru  $p$  (wszystkie wyniki testów umieszczono w dokumentacji technicznej). Reprezentantów algorytmu genetycznego dla danej wartości parametru  $p$  dobierano oddzielnie dla wartości uzyskanego komitetu oraz czasu działania. Dla danego parametru  $p$  wybierano najwyższą uzyskaną wartość komitetu oraz najkrótszy czas spośród tych wyników algorytmu genetycznego, które były co najmniej tak samo dobre jak dla wyniku algorytmu zachłannego. Zestawienia wartości komitetów oraz czasów działania umieszczono w poniższych tabelach.

parametr $p$	algorytm zachłanny	algorytm genetyczny
1	14364	14364
5	1906.82	1906.82
50	1441.4	1446.44

Tabela 7: Zestawienie wartości komitetów

parametr $p$	algorytm zachłanny	algorytm genetyczny
1	2.1	10.22
5	10.87	27.65
50	11.64	13.27

Tabela 8: Zestawienie czasów działania

Uzyskano wyniki, w których dla każdej wartości parametru  $p$  algorytm genetyczny dla pewnego zestawu parametrów działania był co najmniej tak dobry jak algorytm zachłanny. Dla niższych wartości parametru  $p$  algorytm genetyczny dla pewnych parametrów uzyskiwał wartości równe wynikom algorytmu zachłannego. Wraz ze wzrostem wartości parametru  $p$  algorytm genetyczny umacniał się w stosunku do algorytmu zachłannego by w końcu zdecydowanie przewyższać jego wartości. Dla  $p = 50$  wyniki dla wszystkich ośmiu kombinacji parametrów działania algorytmu genetycznego były lepsze od wyniku algorytmu zachłannego. Czas działania algorytmu zachłannego był krótszy od czasu trwania reprezentantów algorytmu genetycznego. Jednak dla wartości parametru  $p$  wynoszącej 50 czasy były bardzo zbliżone. Czasy działania algorytmu genetycznego i zachłannego są coraz bliżej siebie wraz ze wzrostem parametru  $p$  ze względu na fakt, że wraz ze wzrostem parametru  $p$  coraz więcej wyników algorytmu genetycznego dla różnych parametrów działania okazuje się być lepszych bądź równych wynikowi algorytmu zachłannego.



## Spis rysunków

1	Diagram komunikacji . . . . .	13
2	Wyniki wyborów w postaci wyświetlonej przez system tabeli . . . . .	19
3	Wyniki wyborów dla $p = 1$ . . . . .	20
4	Wyniki wyborów dla $p = 5$ . . . . .	20
5	Wyniki wyborów dla $p = 20$ . . . . .	21
6	Średnie i mediany pokrycia się zwycięskich komitetów . . . . .	23

## Materiały źródłowe

- [1] E.Elkind, P.Faliszewski, P.Skowron. *Properties of Multiwinner Rules*, 2015
- [2] *Python 2.7* documentation, <https://docs.python.org/2/>
- [3] *Django* documentation, <https://docs.djangoproject.com/en/1.10/>
- [4] I. Sommerville, *Inżynieria oprogramowania/Ian Sommerville*; z ang. przeł. Krzysztof Stencel, WNT 2003