Dokumentacja projektu

‘ Znajdowanie optymalnego miejsca parkingowego w oparciu w ważony MaxSat Solver’

Studio Projektowe I

Twórcy : Bartosz Biegun , Paweł Hanzlik

Opiekun: prof. Radosław Klimek

05.05.2021

Spis treści

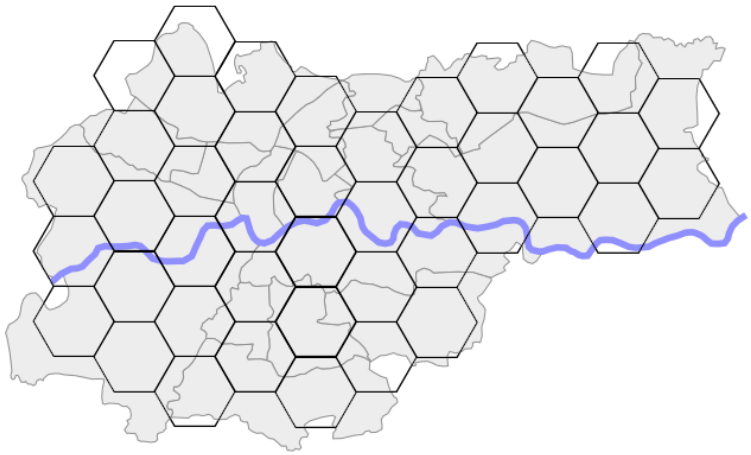
1. Cel projektu

Celem projektu jest zaimplementowanie aplikacji wykorzystującej solver typu MaxSat do znajdowania miejsc parkingowych dla klientów wypożyczalni samochodów z uwzględnieniem popytu na następne wypożyczenia w danym sektorze miasta.
Aplikacja będzie symulowała obszar danego miasta podzielonego na wyznaczone strefy, wewnątrz których jest pewna liczba samochodów dostępnych do wypożyczenia. Każda strefa będzie mieć przewidywany popyt na samochody, na który aplikacja będzie odpowiadać przekierowując samochody do danej strefy gdy są tam potrzebne. Użytkownik będzie posiadał możliwość utworzenia zapytania odpytującego serwer o miejsce parkingowe w pobliżu pewnej lokacji lub uruchomić prostą symulację generującą wiele podobnych zapytań oraz modyfikującą stan bazy w zależności od pory dnia.

2. Zasada działania

a. Strefy

Miasto zostanie podzielone na heksagonalne strefy w celu zarządzania popytem i podażą samochodów na zróżnicowanym obszarze.



W celu optymalnego rozlokowania klientów dla każdej strefy jest regularnie obliczany współczynnik zajętości:

$$a_n = \frac{S_n}{D_n}$$

gdzie:

a_n- Współczynnik zajętości

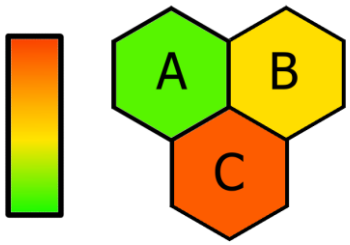
S_n- Suma wag klientów chcących wypożyczyć samochód w odległości do 1 km od środka strefy

D_n- Suma wag wolnych samochodów w odległości do 1 km od środka strefy

Wagi klientów i miejsc parkingowych maleją wraz z odległością od środka strefy co odwzorowuje możliwość że klienci na skraju strefy zostaną przypisani do miejsc parkingowych w sąsiadującej strefie

Aplikacja będzie manipulować klientami chcącymi zaparkować w pobliżu strefy a przez to sumą wag wolnych samochodów aby utrzymać współczynnik zajętości zbliżony do 1

Na przykład:



Samochody ze strefy A (O niskim współczynniku zajętości – samochodów jest za dużo) będą przekierowywane do strefy C (O wysokim współczynniku zajętości – samochodów brakuje)

b. Solver

Problem spełnialności to koncept związany z logiką matematyczną, zostanie on wykorzystany do rozwiązania problemu zarządzania wypożyczanymi samochodami w Krakowie.

Z problemem SAT mamy do czynienia gdy mając formułę zdaniową chce się określić, czy istnieje podstawienie wartościami ‘0’ i ‘1’ pod zmienne zdaniowe, by formuła była spełniona.

Problemy Max-Sat składają się z ważonych klauzul połączonych koniunkcjami :

$(\neg p1 \vee \neg p2) \wedge (\neg p1 \vee p3) \wedge (\neg p1 \vee \neg p3)$

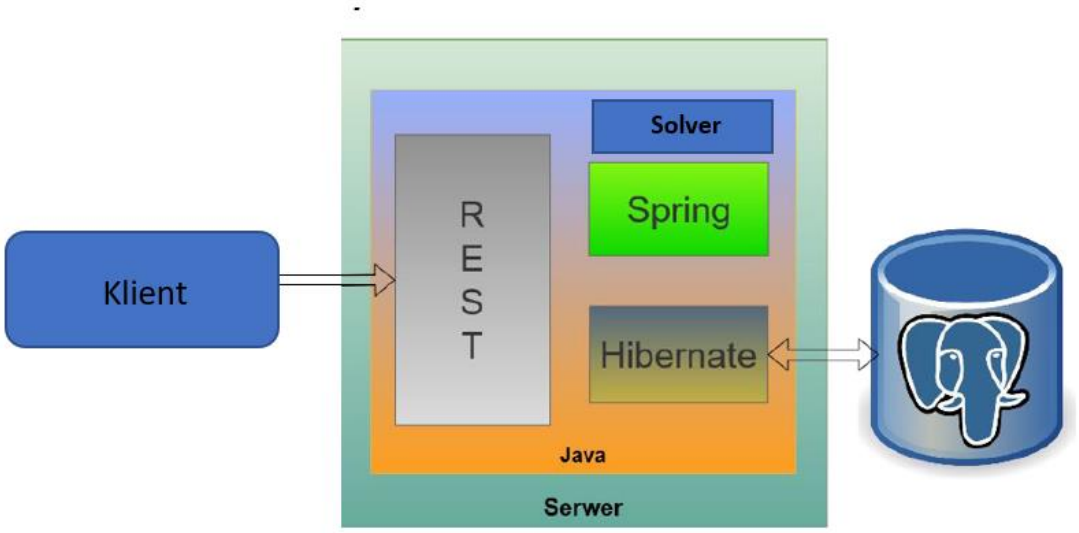
Ponieważ nie zawsze da się spełnić wszystkie klauzule, solver Max Sat znajduje rozwiązanie w którym największa liczba klauzul jest spełniona

Z problemem Max-SAT to rozszerzenie problemu SAT w taki sposób, aby w przypadku gdy nie da się dobrać wartości tak, aby spełniona była cała formuła dobiera się je tak, by zmaksymalizować ilość spełnionych formuł.

Ważony Max-SAT to kolejne rozszerzenie, dodające tym razem odpowiadające wagi każdej z klauzul i uwzględnienie ich a procesie rozwiązywania w taki sposób aby suma wag niespełnionych klauzul była jak najmniejsza.

Aby dobrać najlepsze dla systemu rozwiązanie będzie on generować formułę w której zdania będą odpowiadać optymalnemu rozlokowaniu samochodów.

3. Architektura systemu



Główny moduł aplikacji zostanie napisany w języku Java z użyciem Spring Framework. Baza danych działać będzie na serwerze PostreSQL, natomiast łączenie jej z projektem realizowane będzie przy użyciu Hibernate. Z zewnątrz klient wysyłać będzie zapytanie obsługiwane przez REST API, które na podstawie danych z bazy oraz obliczeń Solvera zwróci wynik.

4. Schemat bazy danych

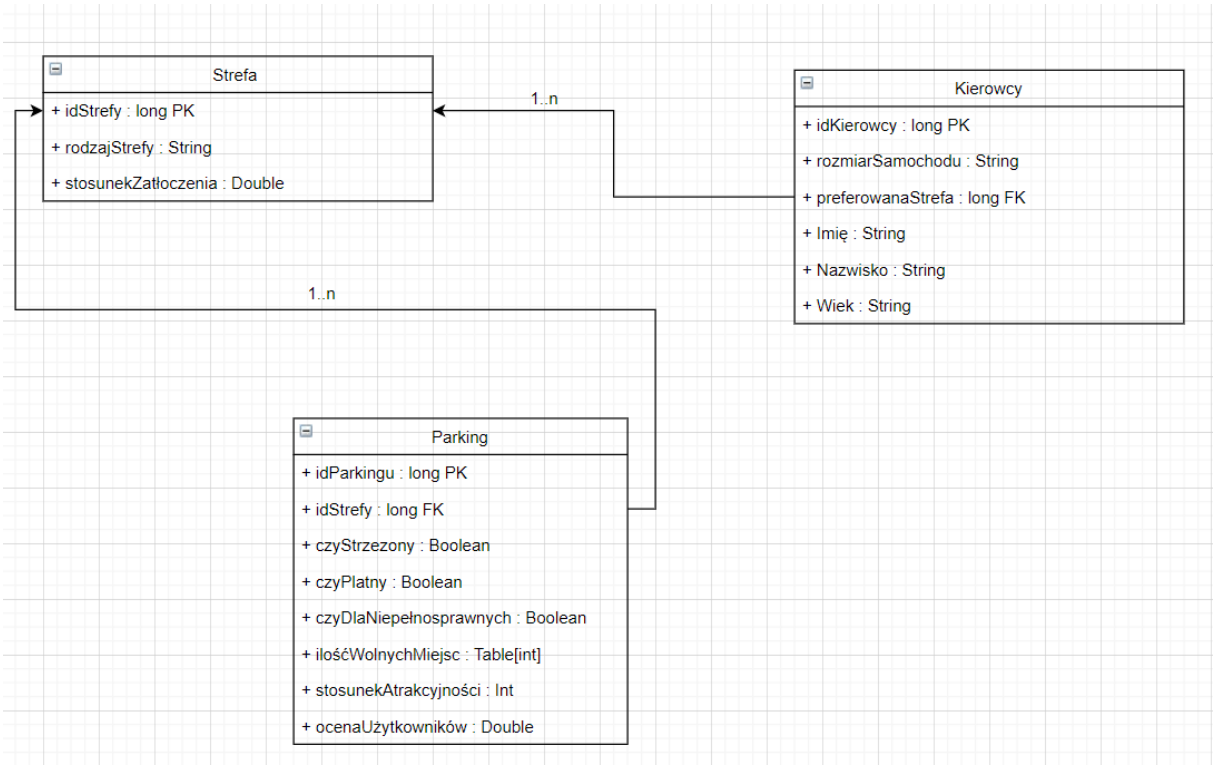


Tabela Strefa oznacza podział miasta na obszary posiadające wartość zatłoczenia w danym momencie oraz rodzaj Strefy tj. obrzeża, centrum, poza miastem, przemysłowa. Zawiera wiele rekordów typu Parking.

Tabela Kierowcy zawiera dane klienta, informacje o rozmiarach ich samochodów oraz preferencjach dotyczących miejsca parkingowego.

Tabela Parking zawiera informacje dotyczące parkingu w danej strefie miasta.

5. Przykłady rekordów w tabelach

Tabela: strefy

idStrefy	rodzajStrefy	stosunekZatłoczenia
1	centrum	0.7
2	centrum	0.8
3	obrzeża	0.5
4	obrzeża	0.4
5	poza miastem	0.1

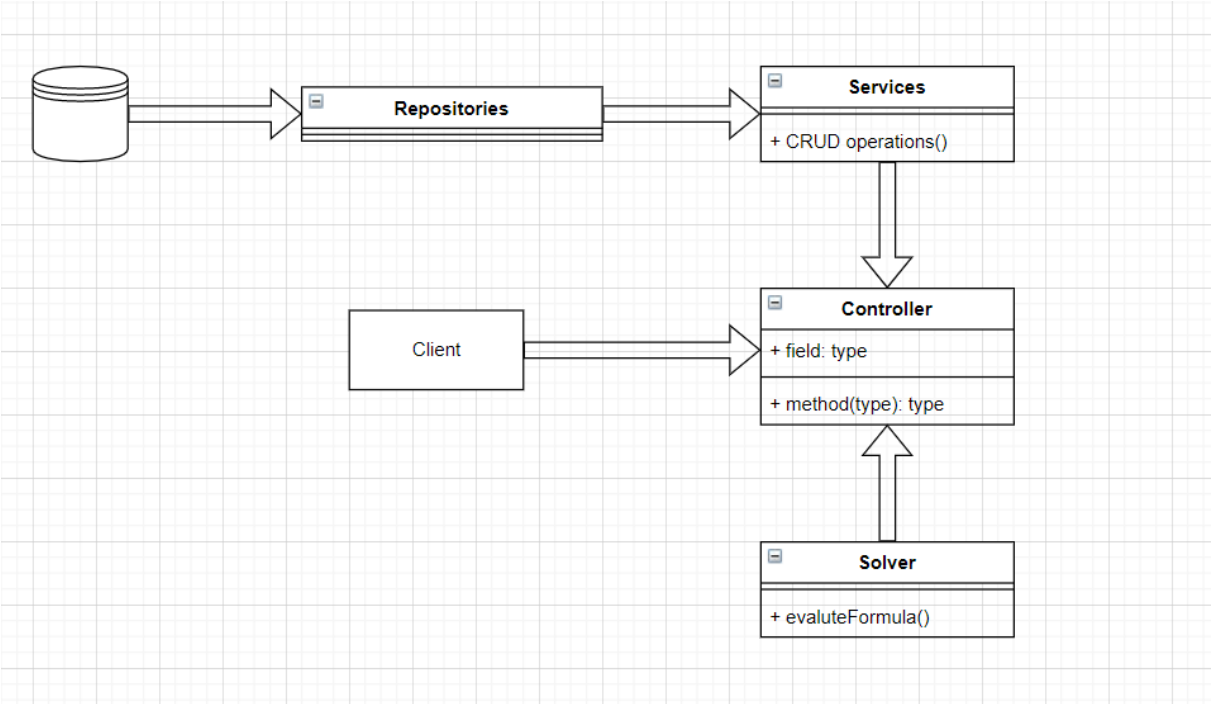
Tabela: kierowcy

idKierowcy	rozmiarSamochodu	preferowanaStrefa	Imię	Nazwisko	Wiek
1	mały	2	Anna	Nowak	34
2	średni	4	Krzysztof	Kowalczyk	23
3	duży	3	Michał	Kowalski	40
4	niepełnosprawność	5	Jan	Nowakowski	56
5	duży	2	Wojciech	Kwiatkowski	42

Tabela: parkingi

idParkingu	idStrefy	czyStrzeżony	czyPłatny	czyDlaNiepełnosprawnych	ilośćWolnychMiejsc	stosunekAtrakcyjności	ocenyUżytkowników
1	2	Tak	Tak	Tak	[50,20]	0.9	5,00
2	2	Tak	Nie	Nie	[100,null]	0.6	4,50
3	1	Nie	NIe	Tak	[150,30]	0.7	4,00
4	4	Tak	Tak	NIe	[30,null]	0.5	4,20
5	3	Tak	Nie	NIe	[60,null]	0.3	4,50
6	5	Nie	Tak	Tak	[40,10]	0.3	3,70

6. Diagram klas



Sekcja Repositories to interfejsy dziedziczące po JPARepository, realizujące połączenie z bazą. Serwisy będą przechowywały metody umożliwiające pobieranie, dodawanie czy też modyfikowanie oraz usuwanie danych z bazy. Solver będzie to zbiór klas, które implementują algorytm Max-Sat solver i na podstawie zapytania klienta zwróci najlepsze miejsce parkingowe. Kontroler będzie przetwarzał zapytanie, przekazywał do solvera i zwracał odpowiedź klientowi.

7. Wykorzystane technologie

- **Java**

  Będziemy używać języka Java w wersji 15

- **Spring**

  Użyjemy SpringBoot'a w wersji 2.4.

- **PostgreSQL**

  Do zarządzania bazą danych. Wersja 13.

- **Hibernate**

Czyli framework realizujący dostęp do danych w bazie. Podstawową rzeczą jaką będziemy wykorzystywać to mapowanie O-R.

- **Lombok**

  Biblioteka Javy udostępniająca adnotacje do tworzenia metod, głównie getterów i setterów.

8. Generowanie klauzul z dostępnych danych

Do przetwarzania danych za pomocą algorytmu Weighted Max-Sat wymagane jest przekształcenie je na zmienne zdaniowe a następnie przyporządkowanie do klauzul.
Każdej klauzuli przypisywana jest waga określająca jak ważne jest spełnienie klauzuli.
Każde dostępne miejsce parkingowe zostaje porównane z klauzulą i przedstawione użytkownikowi. Lista dostępnych miejsc zostaje posortowana po ilości spełnionych klauzul tak aby najbardziej dopasowane miejsca znalazły się na górze listy.

c. Określenie zmiennych zdaniowych

Na podstawie bazy danych wyróżnimy 6 zmiennych zdaniowych:

- S1 – Parking jest w odległości 500 m od klienta
- S2 – Parking znajduje się w strefie o niskim współczynniku zapotrzebowania
- S3 – Parking znajduje się w strefie o średnim współczynniku zapotrzebowania
- S4 – Parking znajduje się w strefie o wysokim współczynniku zapotrzebowania
- S5 – Parking znajduje się w tej samej strefie co klient

d. Określenie klauzul

Analizując dane o kliencie możemy oszacować jego preferencje i dopasować do nich zmienne zdaniowe

Analizując dane o kliencie możemy oszacować jego preferencje i dopasować do nich zmienne zdaniowe

- U1 – Strefa w której znajduje się klient ma niski współczynnik zapotrzebowania
- U2 – Strefa w której znajduje się klient ma średni współczynnik zapotrzebowania
- U3 – Strefa w której znajduje się klient ma wysoki współczynnik zapotrzebowania

U1 
S1 (90)
¬S2 (40)
S4 (70)

U2 
S1 (90)
¬S2 (20)
S4 (45)

U3 
S1 (90)
¬S2 (60)
S5 (70)

e. Interpretacja wyników

Algorytm iteruje po wszystkich miejscach parkingowych i wyodrębnia te z największą ilością spełnionych klauzul, a następnie przedstawia je użytkownikowi jako listę.