

Pracownia Problemowa I

Odtworzenie algorytmu Eclat na dużym zbiorze danych

Emil Kowalczyk
230178

Joanna Ściebura
204169

Marcel Młodziński
217119

Maciek Blankenburg
202475

20 czerwca 2021

Spis treści

1	Wstęp	3
2	Organizacja pracy	3
	Slack	3
	Trello	3
	Github	3
	Dropbox	6
3	Opis działania algorytmu	8
4	Dane wejściowe	9
5	Wyniki	10
6	Podsumowanie	10
7	Bibliografia	10

1 Wstęp

Niniejsza praca omawia zagadnienie wydajności jednego z algorytmów „Data Miningu”, służącego do wydobywania danych za pomocą baz danych. Wykorzystywany w analizie koszykowej, czyli metodzie, która dla tworzy dla zbioru danych zestaw opisujących go przybliżonych reguł asocjacyjnych, tj. powiązań i skojarzeń pomiędzy konkretnymi wartościami zmiennych. Metody asocjacyjne są bardzo przydatne przy operowaniu na dużych zbiorach danych, gdyż przetwarzają one dane w taki sposób, żeby móc wyciągnąć jak najwięcej relacji, które w dziedzinach takich jak sklepy są bardzo przydatne. Przykładowo, pomagają one w ustawianiu produktów tak, żeby klient przy zakupie produktu „X”, zauważył powiązany produkt „Y” i przy okazji go zakupił.

2 Organizacja pracy

W celu jak najlepszego zorganizowania pracy, korzystaliśmy z wielu narzędzi, które na co dzień użytkowane są w branży „IT” i w przypadku ich potencjalnego braku, wiele współczesnych firm mogło czuć się w swoich działaniach „jak bez jednej ręki”.

Slack

Slack traktowany był przez nas jako główny kanał komunikacyjny, jest to darmowa usługa oparta na chmurze, która dzięki posiadaniu wielu narzędzi i usług, używana jest przez największe korporacje informatyczne w naszym kraju. Dzięki możliwości podpinania wielu zewnętrznych aplikacji oraz dzięki funkcjom takim jak „Thread”, zapewnia ona poprawienie organizacji pracy pomiędzy grupą.

Poniżej widnieje zrzut ekranu z panelu głównego aplikacji:

Dzięki wyżej wspomnianym funkcjom Slacka, mogliśmy skonfigurować aplikację tak, żeby ta, mogła korzystać z zewnętrznych aplikacji takich jak Trello czy Github, które będą opisane niżej.

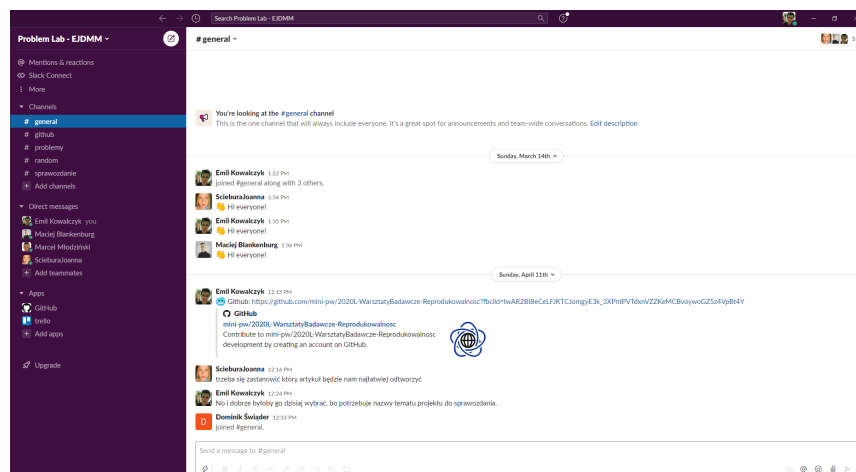
Funkcjonalności podpięcia zewnętrznych aplikacji, umożliwiały śledzenie tego, co dzieje się na zewnętrznych aplikacjach, jednocześnie nie musząc do nich wchodzić.

Trello

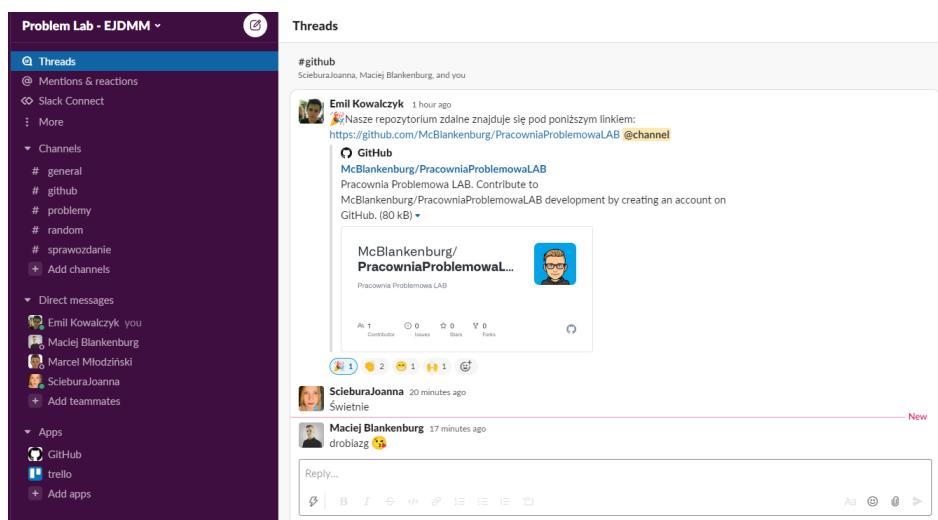
Trello to aplikacja, która pomogła nam w zarządzaniu jednostkami w zespole, jest to internetowa aplikacja do tworzenia list w stylu Kanban. Dzięki niej mogliśmy stworzyć tablicę, na której rozpisane były cele jak i obowiązki każdego z nas.

Github

Github – To nieodłączny element pracy w grupie, czyli hostingowy serwis internetowy, przeznaczony do przechowywania projektów, wykorzystujących system



Rys. 2.1: Zrzut ekranu z aplikacji Slack

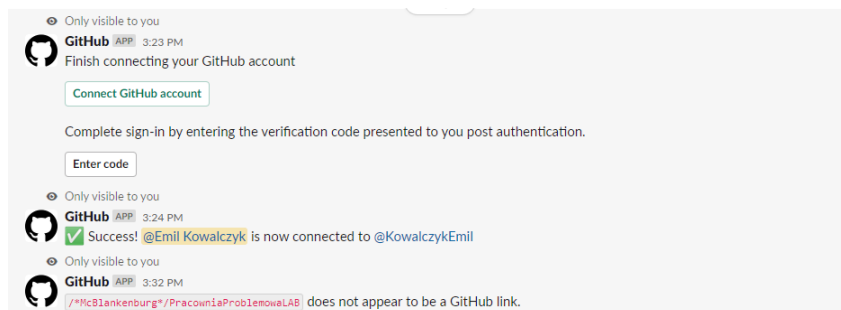


Rys. 2.2: Zrzut ekranu z aplikacji Slack przedstawiający komunikację zespołu

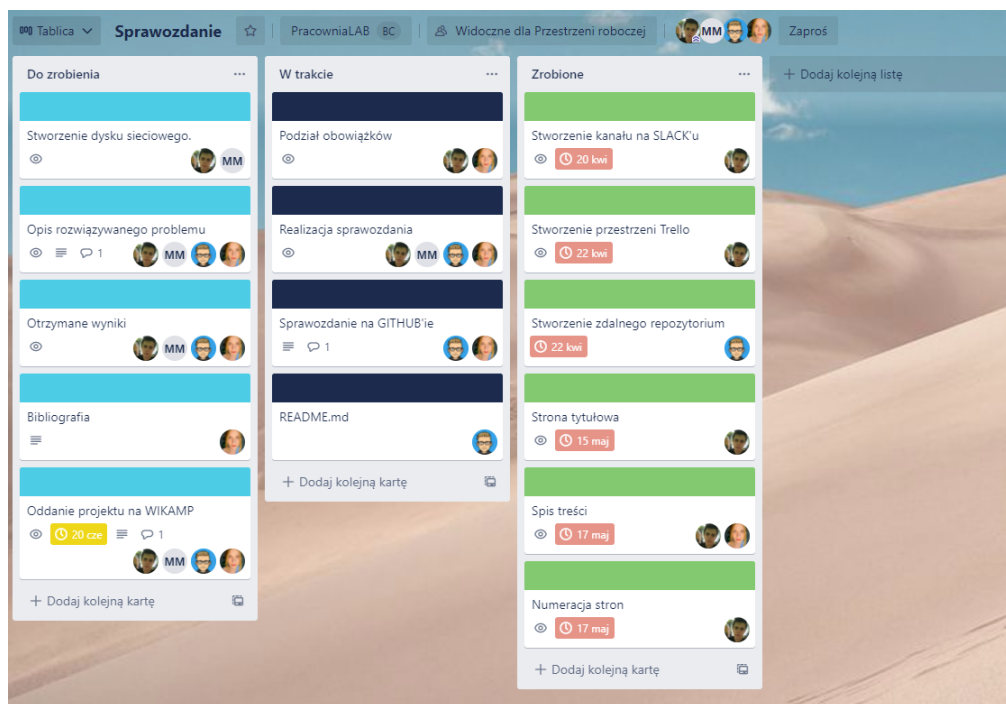
kontroli wersji GIT.

To właśnie na Githubie, przechowywaliśmy dokumentację, którą bez problemu każdy z nas mógł sobie pobierać i wprowadzać swoje zmiany w czasie rzeczywistym, tak, że przy poprawnym korzystaniu z systemu kontroli wersji, każdy mógł pracować na świeżo zaktualizowanej pracy, zachowując przy tym całą historię dokumentacji.

Poniżej widać, że użytkowanie GIT'a, w połączeniu ze zdalnym repozytorem, to naprawdę prosta rzecz i żeby móc bez problemu korzystać z takiej



Rys. 2.3: Zrzut ekranu z konfiguracji Slacka z Githubem



Rys. 2.4: Zrzut ekranu z aplikacji Trello

konfiguracji, wystarczy do tego obsługa kilku komend. Dlatego jest to tak popularna rzecz w obecnych czasach.

Dzięki wyżej wspomnianemu githubowi, mogliśmy umieszczać zmiany w dokumentacji, w taki sposób, że każda osoba z grupy, niezależnie od tego gdzie się znajdowała, mogła te zmiany widzieć oraz z nich korzystać pobierając sobie obecną wersję projektu np.: dzięki korzystania z funkcji **git pull**.

```

Emil@DESKTOP-85M90R6 MINGW64 ~/Desktop/PracowniaProblemowa/Sprawozdanie/PracowniaProblemowaLAB (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  PracowniaProblemowa_EK_JS_MM_MB.tex

nothing added to commit but untracked files present (use "git add" to track)

Emil@DESKTOP-85M90R6 MINGW64 ~/Desktop/PracowniaProblemowa/Sprawozdanie/PracowniaProblemowaLAB (master)
$ git add .

Emil@DESKTOP-85M90R6 MINGW64 ~/Desktop/PracowniaProblemowa/Sprawozdanie/PracowniaProblemowaLAB (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

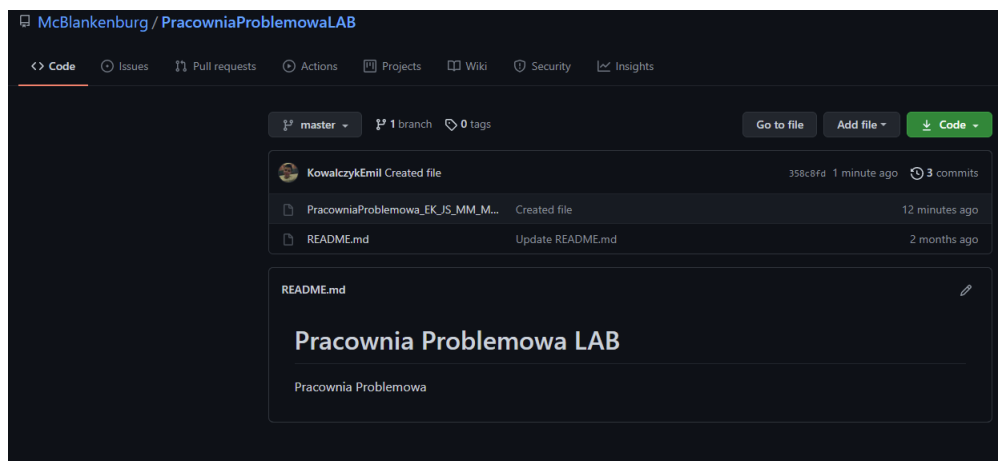
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   PracowniaProblemowa_EK_JS_MM_MB.tex

Emil@DESKTOP-85M90R6 MINGW64 ~/Desktop/PracowniaProblemowa/Sprawozdanie/PracowniaProblemowaLAB (master)
$ git commit -m "Created file" PracowniaProblemowa_EK_JS_MM_MB.tex
[master 358c8fd] Created file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 PracowniaProblemowa_EK_JS_MM_MB.tex

Emil@DESKTOP-85M90R6 MINGW64 ~/Desktop/PracowniaProblemowa/Sprawozdanie/PracowniaProblemowaLAB (master)
$ git push -u origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 310 bytes | 310.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/McBlankenburg/PracowniaProblemowaLAB.git
56b0fb4..358c8fd master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

```

Rys. 2.5: Zrzut ekranu z konsoli GIT'a

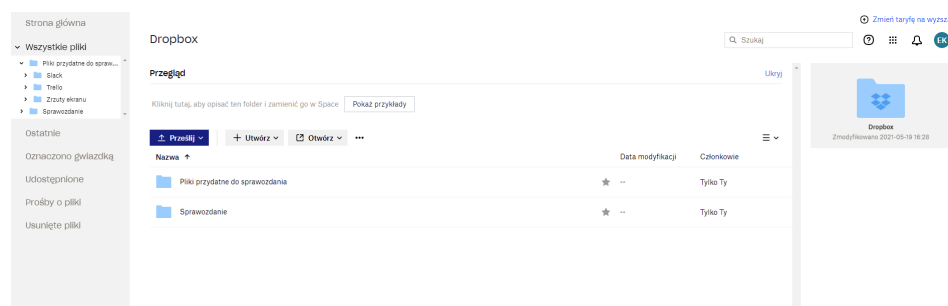


Rys. 2.6: Zrzut ekranu ze zdalnego repozytorium

Dropbox

Dropbox – usługa pozwalająca nam przechowywać pliki w przestrzeni chmurowej. Była przez nas rzadko wykorzystywana, jednak wszelakie zrzuty ekranu, były

umieszczane właśnie tam, dzięki czemu nasze dyski na lokalnych komputerach, nie muszą przechowywać zbędnych danych, które mogą być przechowywane w chmurze, dodatkowo mając dodatkową funkcjonalność w postaci możliwości pobierania tych danych przez każdą osobę z zespołu w dowolnym momencie.



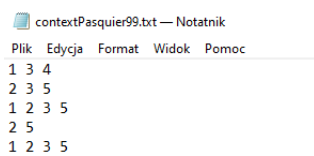
Rys. 2.7: Zrzut ekranu z aplikacji dropbox

3 Opis działania algorytmu

Eclat – jest to algorytm do znajdowania częstych zestawów produktów w transakcji lub bazie danych. Jest to jedna z najlepszych metod uczenia się reguł asocjacyjnych. Algorytm ten, jest używany do generowania częstych zestawów przedmiotów w bazie danych.

W porównaniu do algorytmu Apriori, w metodzie zaproponowanej przez Zaki'ego nastąpiła znacząca redukcja narzutu związanego z operacjami wejścia/wyjścia. Eclat wykonuje trzy odczyty z bazy danych, natomiast algorytmy z rodziny Apriori muszą przeszukiwać dbazę danych tyle razy, ile wynosi maksymalna długość zbioru częstego. Ponadto, już na początku wykonywania algorytmu Eclat uzyskujemy niezależność pomiędzy procesami. Dzięki czemu nie jest potrzebna synchronizacja w trakcie generacji kolejnych kandydatów na zbiory częste. Algorytm Eclat posiada również istotne wady, główne narzuty związane z komunikacją wynikają z potrzeby ustalenia globalnych tid-list na początku algorytmu. W większości przypadków udaje się je zamortyzować w kolejnych iteracjach algorytmu. W trakcie wykonywania algorytmu dla różnych danych może dochodzić do nierównego obciążenia procesorów. Aby tego uniknąć liczba procesów powinna być dużo mniejsza niż liczba klas równoważności dla zbiorów częstej o długości 2. Aby osiągnąć dobrą wydajność algorytmu dla dowolnych danych może być konieczne przygotowanie strategii rozdziału klas równoważności oraz równoważenie obciążeń.

4 Dane wejściowe

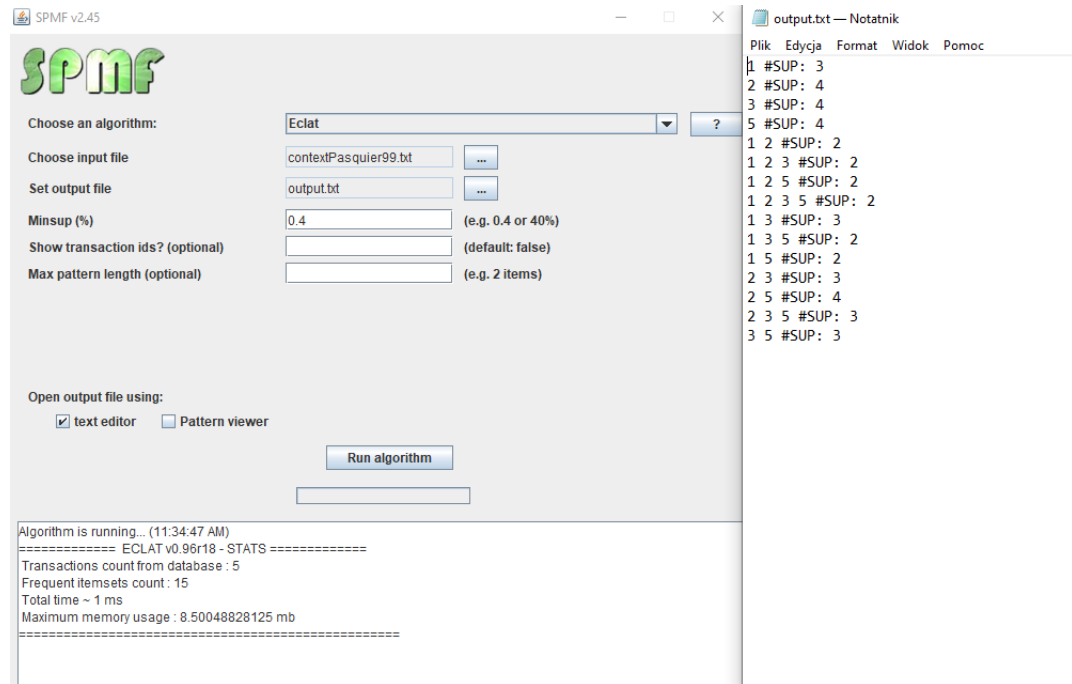


A screenshot of a Notepad window titled "contextPasquier99.txt — Notatnik". The window contains five lines of text, each representing a row of a game board. The text is as follows:

```
Plik  Edycja  Format  Widok  Pomoc
1 3 4
2 3 5
1 2 3 5
2 5
1 2 3 5
```

Rys. 4.1: Dane wejściowe

5 Wyniki



Rys. 5.1: Wyniki działania aplikacji

6 Podsumowanie

Algorytm Eclat posiada również istotne wady, główne narzuty związane z komunikacją wynikają z potrzeby ustalenia globalnych tid-list na początku algorytmu. W większości przypadków udaje się je zamortyzować w kolejnych iteracjach algorytmu. W trakcie wykonywania algorytmu dla różnych danych może dochodzić do nierównego obciążenia procesorów. Aby tego uniknąć liczba procesów powinna być dużo mniejsza niż liczba klas równoważności dla zbiorów częsty o długości 2. Aby osiągnąć dobrą wydajność algorytmu dla dowolnych danych może być konieczne przygotowanie strategii rozdziału klas równoważności oraz równoważenie obciążeń.

7 Bibliografia

Algorytm Eclat posiada również istotne wady, główne narzuty związane z komunikacją wynikają z potrzeby ustalenia globalnych tid-list na początku algorytmu. W większości przypadków udaje się je zamortyzować w kolejnych iteracjach

algorytmu. W trakcie wykonywania algorytmu dla różnych danych może dochodzić do nierównego obciążenia procesorów. Aby tego uniknąć liczba procesów powinna być dużo mniejsza niż liczba klas równoważności dla zbiorów częsty o długości 2. Aby osiągnąć dobrą wydajność algorytmu dla dowolnych danych może być konieczne przygotowanie strategii rodziału klas równoważności oraz równoważenie obciążeń.

1. Nyćkowiak Jędrzej, Leśny Jacek (red.): Badania i Rozwój Młodych Naukowców w Polsce: Nauki techniczne i inżynierijskie. Cz. 2, 2019, Poznań, Młodzi Naukowcy, 196 s., ISBN 978-83-66392-43-4
2. trójpasmowe z luminoforami wąskopasmowymi
3. z luminoforami wielopasmowymi