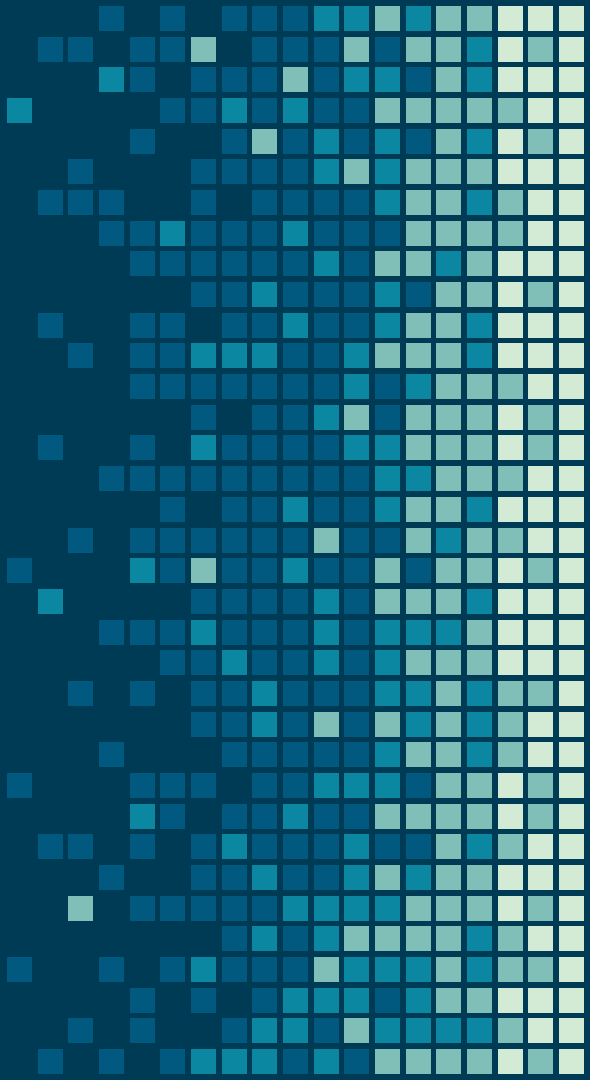
Hash like you never done before



Warszawa, 25.06.2018
Michał Nowotka | github.com/mnowotka | mmmnow@gmail.com

Cześć!

Michał Nowotka

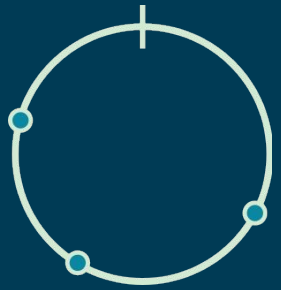
Backend Team Lead at YouGov.

github.com/mnowotka

Czym jest haszowanie?

- Obliczanie kryptograficznej funkcji skrótu.
- Mała zmiana w dokumencie daje kompletnie inną wartość funkcji skrótu.
- Minimalizacja kolizji, użyteczna przy implementacji zbiorów, sum kontrolnych itd.

Oprócz klasycznych funkcji mieszających, istnieją inne o dodatkowych cechach lub przeciwnym celu.



Consistent Hashing

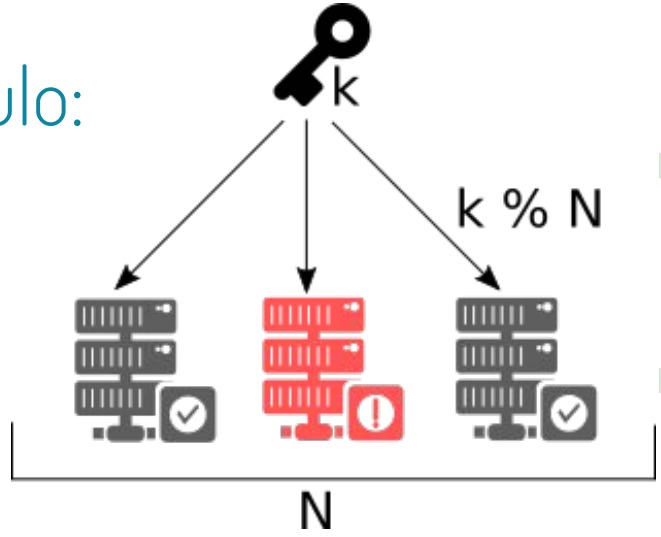
Odporność na zmianę rozmiaru tablicy haszującej.

CONSISTENT HASHING

Rozwiązuje problem load balancingu:
Jak rozłożyć webowy cache na wiele serwerów?

Rozwiązanie naiwne - modulo:

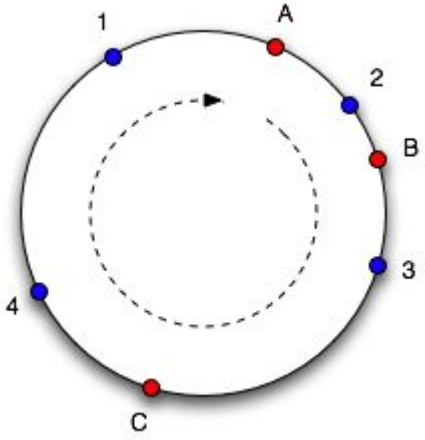
- Dla *N* serwerów i klucza *k*, umieść klucz na serwerze *k % N*.
- Dodanie (lub usunięcie) serwera jest równoznaczne z unieważnieniem całego cache'u.
- Czy możemy sobie na to pozwolić?



CONSISTENT HASHING

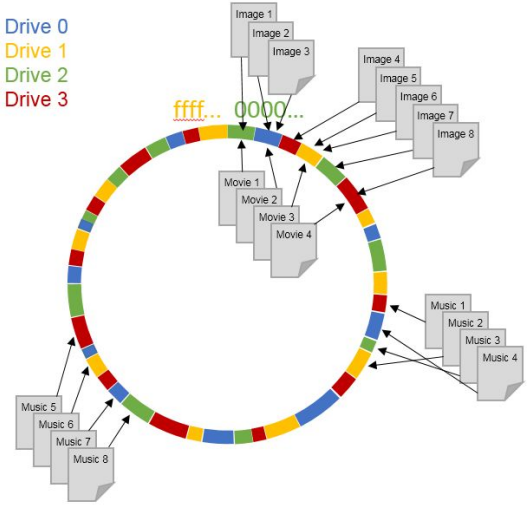
Spójne haszowanie przy zmiennej wielkości tablicy haszującej, minimalizacja przenoszonych kluczy.

- Zasoby (klucze) są punktami na okręgu.
- Serwery też są (kilkoma) punktami na okręgu.
- Dany zasób jest przypisany do najbliższego serwera, zgodnie z ruchem wskazówek zegara
- Przy zmianie wielkości puli serwerów, średnio tylko K/N kluczy musi zostać przeniesionych.



CONSISTENT HASHING – DEMO

Symulujemy serwer z dyskami o pojemności 2 TB każdy. Zapiszemy na nich milion plików o maksymalnej o losowej wielkości zadanej rozkładem normalnym.



- Łatwo zauważyć, że system plików jest zajęty jedynie w 80%.
- Mimo to, istnieją dyski zajęte w niemal 95%.
- Czy można napisać procedurę, która przeniesie dane tak, by dyski były zajęte w max. 90%?
- Całkowita ilość przeniesionych danych powinna być mniejsza niż połowa dysku.




Locality Sensitive Hashing

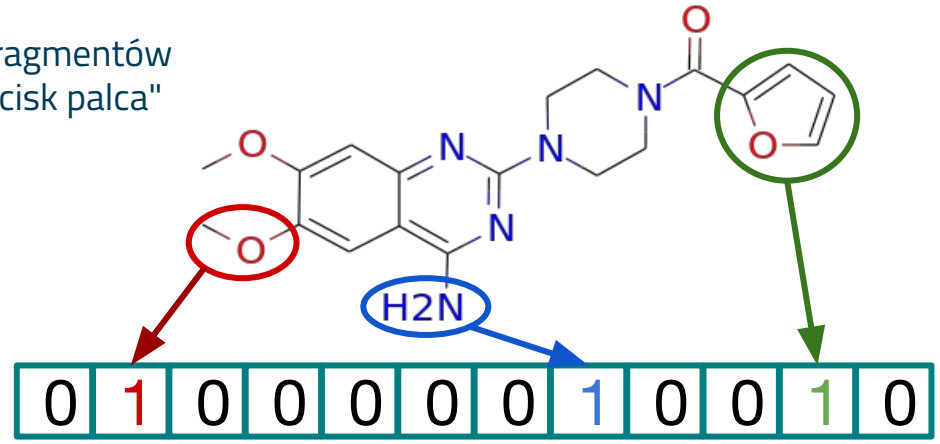
Maksymalizacja prawdopodobieństwa kolizji dla podobnych elementów zbioru.

LSH – najważniejsze cechy

- LSH haszuje wejściowe elementy w ten sposób, że podobne elementy z dużym prawdopodobieństwem wpadają do tych samych "kubełków".
- Redukuje ilość wymiarów dla wielowymiarowych danych.
- Ma wiele wspólnego z klasteryzacją.
- Bardzo popularne w rozpoznawaniu dźwięku (Shazam).

Podobieństwo cząstek – jak to działa?

Obecność pewnych fragmentów cząstki tworzy jej "odcisk palca"



"rzadki" wektor bitów:

0	1	0	0	0	0	0	1	0	0	1	0

Teraz można zdefiniować podobieństwo wektorów:

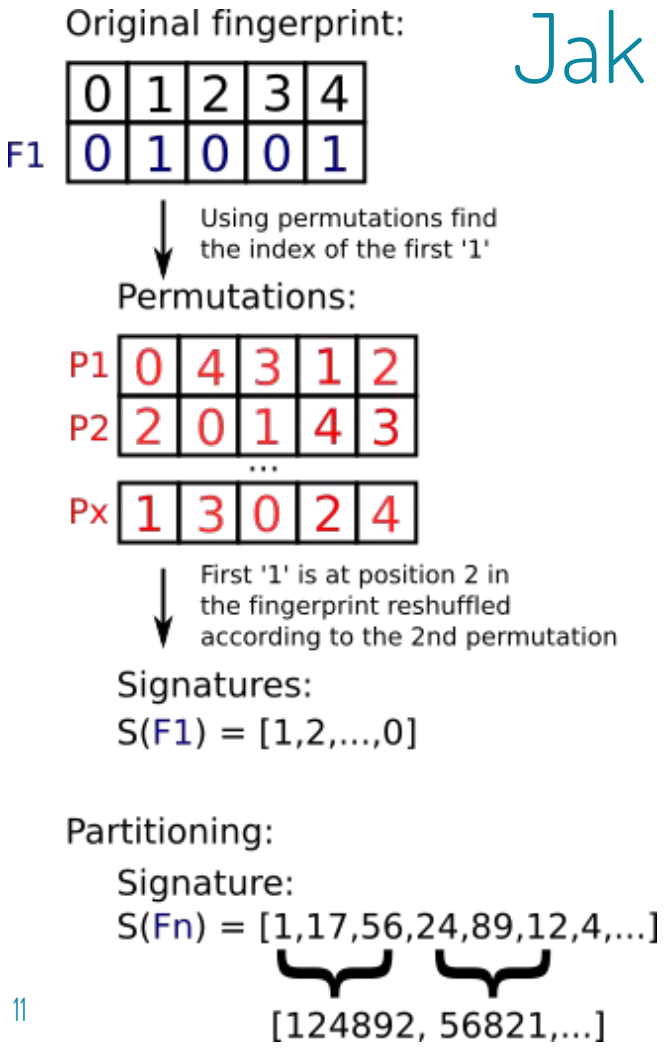
Tanimoto score (Przecięcie przez unię)

$$T(a,b) = \frac{N_{a,b}}{N_a + N_b - N_{a,b}}$$

1	0	1	0
0	1	1	0

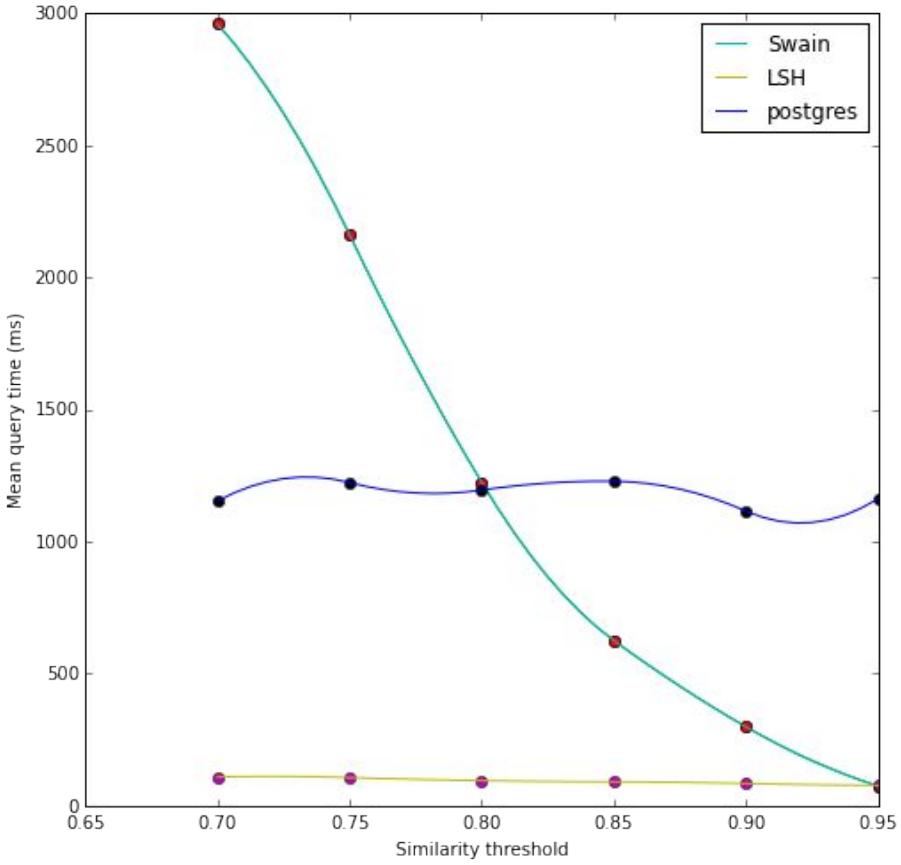
⇨ $\frac{1}{2 + 2 - 1} = \frac{1}{3}$

Jak działa LSH?



- Wygeneruj X losowych permutacji o długości wektora.
- Dla każdego wektora, znajdź indeks pierwszej jedynki dla każdej permutacji.
- Podziel tak powstałą sygnaturę na Y fragmentów i zapisz każdy fragment w jednej z Y map.
- Dla cząstki będącej zapytaniem, wygeneruj jej sygnaturę i każdy z fragmentów użyj jako zapytania do jednej z Y map.

LSH – efekty


Swain
LSH
postgres
Mean query time (ms)
Similarity threshold
3000
2500
2000
1500
1000
500
0
0.65
0.70
0.75
0.80
0.85
0.90
0.95


Join YouGov!
www.linkedin.com/company/yougov/jobs/
YouGov®

Dziękuję!

Pytania?

Znajdź mnie:

mmmnow@gmail.com

michal.nowotka@yougov.com

Bibliografia

- Duplicate songs detector via audio fingerprinting
- LSH-based similarity search in MongoDB
- Understanding consistent hashing