

Rozdział I

Zastosowanie dolnoprzepustowej filtracji sygnałów do ciągłego pomiaru obciążenia systemu zarządzania bazą danych

Streszczenie. Wraz ze znacznym wzrostem złożoności współczesnych systemów baz danych, techniki automatycznej optymalizacji nabrały szczególnego znaczenia. Systemy optymalizacji ciągłej na bieżąco dopasowują parametry systemu do zmieniającego się obciążenia. Podstawowym elementem każdego systemu ciągłego strojenia jest moduł analizujący bieżące obciążenie systemu. W artykule opisano algorytm analizy bieżącego obciążenia systemu, pozwalający szacować takie parametry obciążenia jak: częstotliwość poszczególnych typów zapytań, średni czas wykonywania zapytań i łączny udział czasu wykonania zapytań. Znane z teorii sygnałów techniki filtracji dolnoprzepustowej pozwoliły uzyskać bardzo niską, stałą złożoność pamięciową i czasową algorytmu przy zachowaniu wystarczającej dokładności pomiarów.

1 Wstęp

Współczesne systemy relacyjnych baz danych posiadają wiele cech umożliwiających strojenie wydajności. Administrator może zwykle ustalać parametry buforowania danych, rozmiary pamięci podręcznej, odpowiedni zestaw indeksów i widoków zmaterializowanych. Szczególnie dobór odpowiedniego zbioru indeksów i widoków zmaterializowanych ma bardzo duży wpływ na wydajność. W złożonych systemach jest to jednak zagadnienie wymagające od administratora szerokiej wiedzy z zakresu implementacji systemów baz danych i optymalizacji zapytań, jak również dogłębnej znajomości aplikacji korzystającej z bazy danych. Ponadto aplikacje zmieniają się w czasie, są dodawane nowe funkcje i nowe rodzaje zapytań, więc nawet początkowo optymalnie skonfigurowany system może z czasem wymagać ponownej optymalizacji.

Rozwiązaniem problemu trudnej, ręcznej optymalizacji jest optymalizacja automatyczna. Algorytmy automatycznego odkrywania indeksów i widoków zmaterializowanych zostały szczegółowo opisane w pracach [1–6]. Algorytmy te potrzebują danych o przewidywanym obciążeniu bazy w postaci listy zapytań i są stosowane na etapie projektowania bazy danych. Lista zapytań może być wygenerowana przez aplikację w czasie testów. Takie statyczne podejście ma jednak tę wadę, że nie pozwala optymalizować systemu w trakcie pracy, wraz ze zmieniającą się charakterystyką obciążenia. Pewne próby rozwiązania tego problemu przedstawiono w [7, 8]. Autorzy tych

Piotr Kołaczkowski

Politechnika Warszawska, Instytut Informatyki, ul. Nowowiejska 15/19, 00-665 Warszawa, Polska
email: pkolacz@ii.pw.edu.pl

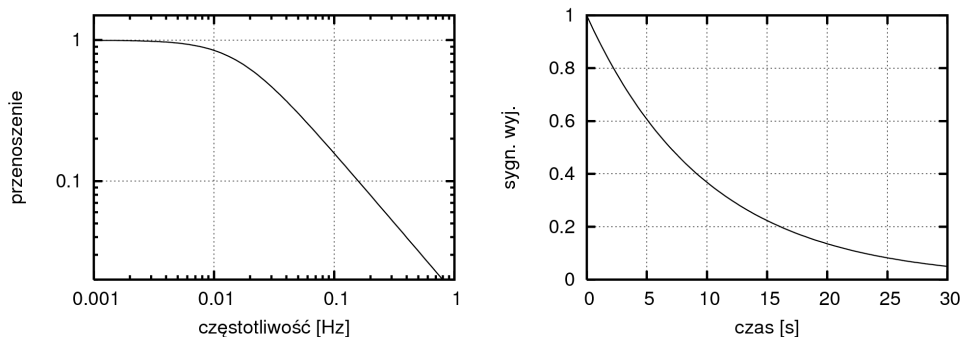
prac skupili się jednak na znalezieniu efektywnych metod modyfikacji zestawu indeksów, pomijając problematykę analizy bieżącego obciążenia bazy danych. W [8] optymalizacja jest dokonywana dla pakietów pewnej arbitralnie przyjętej liczby zapytań, w [7] – dla każdego zapytania osobno.

Metoda zaprezentowana w tym artykule pozwala na bieżąco szacować obciążenie systemu. Na podstawie zebranych statystyk można określić, które typy zapytań były wykonywane najczęściej, które wykonywały się najdłużej oraz które łącznie w ostatnim czasie najbardziej obciążały system.

Należy zwrócić uwagę na to, że wprawdzie problem znajdowania częstych zapytań w strumieniu jest dobrze opisany w literaturze [10–12], to jednak algorytmy te nie uwzględniają efektu starzenia się danych. Statystyki odnoszące się do późniejszych zapytań są w przedstawionym tu algorytmie ważniejsze, niż statystyki zapytań dawnych. Z kolei od algorytmów opartych o okna przesuwne [9] lub analizę pakietów zapytań o stałej długości [8, 9], nasze rozwiązanie różni się brakiem konieczności przechowywania historii zapytań lub jej fragmentów i tym samym znacznie mniejszą złożonością pamięciową.

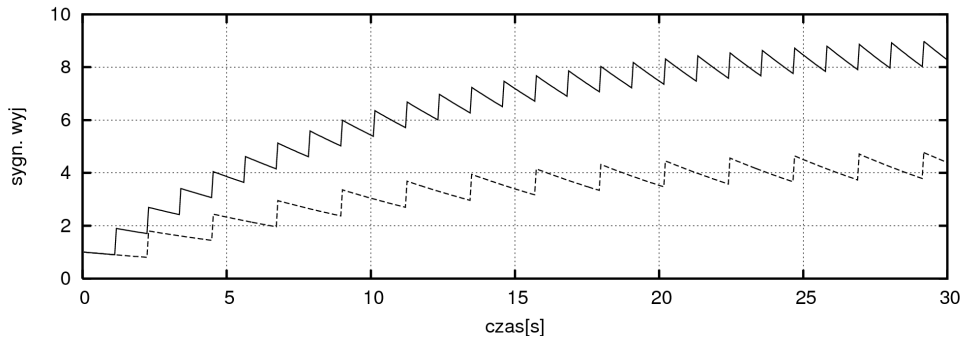
2 Podstawy teoretyczne

Filtry dolnoprzepustowe są dobrze opisane przez teorię sygnałów. Zarówno realizacje analogowe jak i cyfrowe stosuje się szeroko w radioelektronice i telekomunikacji. Opis własności filtrów sygnałów można znaleźć praktycznie w każdym podręczniku do teorii sygnałów np. [13]. W algorytmie tu zaprezentowanym wykorzystywana jest najprostsza filtracja dolnoprzepustowa (FD) pierwszego rzędu.



Rys. 1. Pasma przenoszenia i odpowiedź impulsowa filtra dolnoprzepustowego pierwszego rzędu o stałej czasowej $\tau = 10$ s

Odpowiedź impulsowa FD jest funkcją malejącą wykładniczo (rys. 1), o wartości początkowej równej 1. Pozwala to wykorzystać filtr do przybliżonego pomiaru częstotliwości. Załóżmy, że na wejście filtru zostanie podany sygnał okresowy składający się z impulsów o postaci delty Diraca, o okresie znacznie mniejszym niż stała czasowa filtru τ . Wtedy, po ustaleniu się stanu równowagi, średnia wartość bezwzględna sygnału wyjściowego filtru będzie w przybliżeniu wprost proporcjonalna do częstotliwości sygnału wejściowego (rys. 2). Natomiast wartość chwilowa będzie odbiegała od wartości średniej, jednak im mniejszy okres sygnału wejściowego w porównaniu ze stałą czasową filtru, tym amplituda tych rozbieżności będzie mniejsza w stosunku do wartości sygnału wyjściowego.



Rys. 2. Sygnał wyjściowy z filtra dolnoprzepustowego pierwszego rzędu o stałej czasowej $\tau = 10$ s, na którego wejście podano sygnał okresowy w postaci nieskończonej amplitudzie (delta Diraca). Górny wykres odpowiada częstotliwości impulsów 1 Hz, dolny dla 0,5 Hz

Stan równowagi zostanie osiągnięty, gdy sygnał wyjściowy w chwili tuż przed każdym impulsem wejściowym będzie jednakowy. Oznaczmy przez $u(t_0)$ wartość sygnału wyjściowego w chwili t_0 tuż przed nadejściem impulsu wejściowego, przez $u(t_1)$ wartość sygnału w chwili t_1 tuż po nadejściu impulsu. Ponieważ odpowiedź impulsowa filtru ma wartość początkową równą 1, można napisać, że:

$$u(t_1) = u(t_0) + 1 \quad (1)$$

Przez czas do nadejścia następnego impulsu w chwili $t_2 = t_0 + \Delta t$ sygnał wyjściowy maleje wykładniczo i w chwili t_2 osiąga wartość:

$$u(t_2) = u(t_1) e^{-\Delta t / \tau} \quad (2)$$

Założywszy, że $\Delta t \ll \tau$, rozwijając prawą stronę równania 2 w szereg Taylora w punkcie $t_2 = t_0$ i zostawiając tylko dwa pierwsze wyrazy, otrzymujemy:

$$u(t_2) \approx u(t_1) (1 - \Delta t / \tau) \quad (3)$$

W stanie równowagi spełniona jest zależność:

$$u(t_2) = u(t_0) \quad (4)$$

Po podstawieniu równań 1 i 3 do równania 4 i rozwiązaniu otrzymujemy:

$$u(t_1) \approx \tau f, \quad (5)$$

gdzie f to częstotliwość impulsów wejściowych. Ponieważ wartość τ jest stała, na podstawie wartości sygnału wyjściowego można obliczyć przybliżoną częstotliwość sygnału wejściowego.

3 Algorytm

W pierwszej kolejności z każdego zapytania usuwane są wartości parametrów. Dzięki temu następują zapytania stają się sobie równoważne:

```
SELECT * FROM table WHERE param = 25;
SELECT * FROM table WHERE param = 30;
```

Z każdym rodzajem zapytania są skojarzone dwa filtry dolnoprzepustowe pierwszego rzędu. Filtr o zmiennej stanu f służy do pomiaru częstotliwości występowania zapytania, zaś filtr o zmiennej stanu l do pomiaru obciążenia bazy danych powodowanego przez to zapytanie. Stan filtrów jest modyfikowany z każdym zapytaniem przez następującą procedurę:

```
time_delta := now() - last_time;
f := f * exp(-time_delta / tau) + 1 / tau;
l := l * exp(-time_delta / tau) + query_duration / tau;
last_time := now();
```

Należy zwrócić uwagę, że zmienne stanu filtrów są dla wygody zwiększane o wartości podzielone przez τ . Dzięki temu zmienna f estymuje częstotliwość zapytań, a zmienna l stosunek czasu wykonywania zapytań danego rodzaju do całego dostępnego czasu. Gdyby było inaczej, zgodnie ze wzorem 5, zmienne stanu filtrów należałoby dzielić przez τ przed każdym odczytem w celu wyświetlenia ich użytkownikowi lub do dalszej interpretacji.

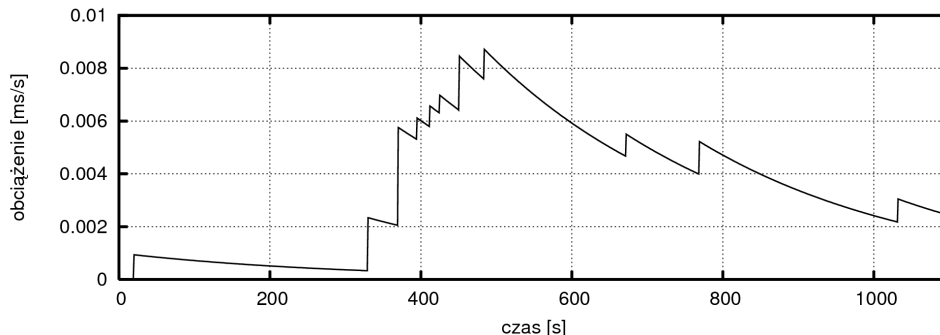
W celu pobrania aktualnych wartości pomiarów w dowolnym czasie, dostępna jest procedura symulująca zmieniający się stan filtrów wraz z upływem czasu, przy zerowym sygnale wejściowym:

```
time_delta := now() - last_time;
f := f * exp(-time_delta / tau);
l := l * exp(-time_delta / tau);
avg_query_duration := load / f;
last_time := now();
```

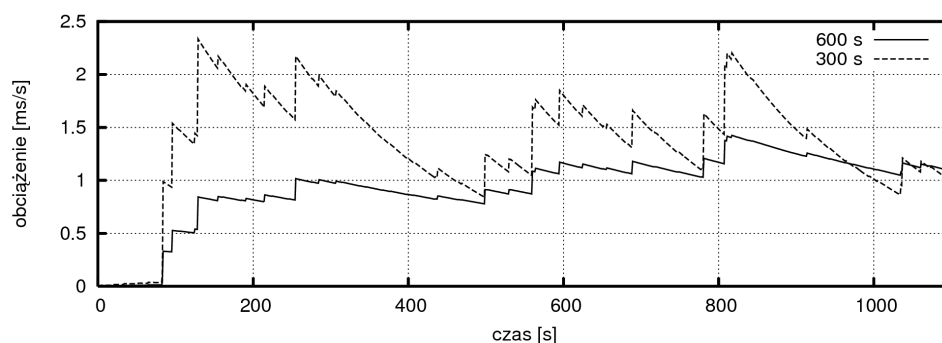
4 Testy

Procedury opisane w poprzednim punkcie zostały zaimplementowane w Javie i wprowadzone do kodu sterownika JDBC. W tym miejscu bowiem są dostępne informacje o czasie wykonywania zapytań. Ze względu na brak narzędzia do automatycznego strojenia bazy danych, przeprowadzona została optymalizacja ręczna. Aplikacja testowa generowała różne zapytania z różną częstotliwością, dochodzącą do kilku tysięcy zapytań na godzinę. Były jednak również zapytania, które wykonywały się rzadziej niż raz na godzinę.

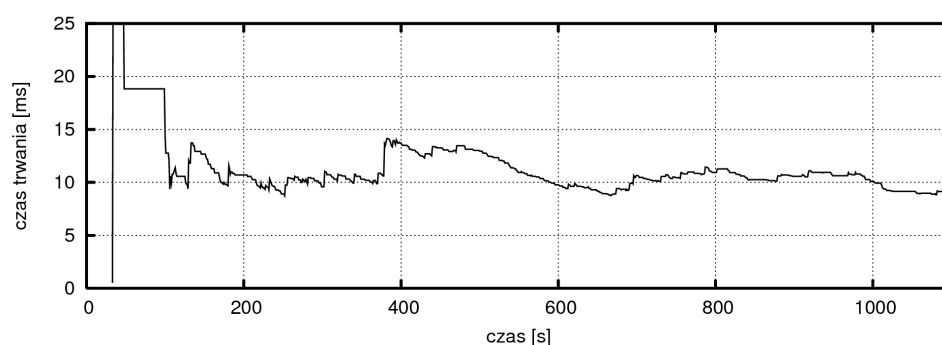
Arbitralnie przyjęto stałą czasową filtrów $\tau = 300$ s, co pozwoliło szybko wykonać optymalizację zapytań częstych, natomiast dla zapytań rzadkich produkowało niestabilne i mało wiarygodne statystyki (rys. 3). Im ustawiono większą stałą czasową, tym statystyki były mniej wrażliwe na chwilowe zmiany obciążenia, jednak również więcej czasu było potrzebne na zoptymalizowanie systemu. Zostało to zobrazowane na rys. 4.



Rys. 3. Wykres obciążenia bazy danych uzyskany dla rzadko występującego zapytania, generującego małe obciążenie



Rys. 4. Wykres obciążenia bazy danych uzyskany za pomocą dwóch filtrów o różnych stałych czasowych



Rys. 5. Wykres średniego czasu wykonania pewnego rodzaju zapytań.

System pozwala również na analizę średniego czasu wykonywania zapytań. Dane te mogą być przydatne do oceny średniego czasu odpowiedzi aplikacji. Przykładowy wykres jest przedstawiony na rys. 5. Pierwsze pomiary charakteryzują się większym rozrzutem (widać wyraźne „schodki”), ponieważ są oparte na niewielkiej ilości danych zebranych od momentu rozpoczęcia pomiarów.

5 Podsumowanie

Przedstawiona w artykule metoda wykazała przydatność do ręcznego strojenia systemu bazy danych. Dzięki zastosowaniu filtracji dolnoprzepustowej (FD), połączono uśrednianie zbieranych pomiarów z równoczesnym stopniowym przedawnianiem starych pomiarów. Dlatego można przewidywać, że opracowana metoda będzie się dobrze sprawdzać w systemach ciągłego strojenia wydajności. Twierdzenie takie wymaga jednak jeszcze potwierdzenia doświadczalnego, co jest obecnie o tyle trudne, że systemy ciągłego strojenia baz danych są nowością [8] i nie zostały jeszcze rozpowszechnione. Dodatkową zaletą zaprezentowanego podejścia jest to, że system ciągłego strojenia, wykorzystujący FD, byłby systemem dynamicznym ze sprzężeniem zwrotnym, które łatwo dałoby się opisać za pomocą rachunku operatorowego. Ułatwiłoby to prawdopodobnie formalną analizę stabilności takiego systemu.

Pewną trudnością w stosowaniu FD jest konieczność określenia stałej czasowej. Z jednej strony istnieje zapotrzebowanie na jak najszybszą reakcję systemu sterującego na

zmieniające się warunki, więc stała ta powinna być niewielka. Niestety z drugiej strony mała stała czasowa powoduje, że założenia przyjęte w rachunkach w podrozdziale 2 nie są już prawdziwe, a dodatkowo system staje się wrażliwy na chwilowe fluktuacje obciążenia. Z tego powodu przyszłe badania będą koncentrować się na metodach automatycznego doboru stałej czasowej, być może różnej dla różnych rodzajów zapytań, by zapewnić akceptowalny poziom błędów statystycznych i dostateczną szybkość działania.

Zapotrzebowanie pamięciowe przedstawionego algorytmu jest wystarczająco niewielkie do zastosowania go do strojenia baz danych. Jednak obecna wersja nie nadaje się do analizy ruchu sieciowego. W tym zastosowaniu złożoność liniowa względem liczby analizowanych kategorii zdarzeń jest zdecydowanie zbyt duża. Trudno byłoby np. zrealizować nawet po jednym filtrze cyfrowym na każdy adres IP dysponując jedynie szybką, niewielką, statyczną pamięcią RAM. Dalsze zmniejszenie złożoności pamięciowej jest przedmiotem przyszłych badań.

Literatura

1. Agrawal S., Chaudhuri S., Narasayya V. R.: Automated selection of materialized views and indexes in SQL databases. VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases, San Francisco, CA, USA, 2000.
2. Bruno N., Chaudhuri S.: Automatic physical database tuning: a relaxation-based approach. Proceedings of the 2005 ACM SIGMOD international Conference on Management of Data, (Baltimore, Maryland, 2005), s. 227 – 238. ACM Press, New York, NY, USA, 2005.
3. Mistry H., Roy P., Sudarshan S., Ramamritham K.: Materialized view selection and maintenance using multi-query optimization. Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (New York, NY, USA, 2001), s. 307 – 318, ACM Press, New York, NY, USA, 2001.
4. Skelley A.: DB2 advisor: An optimizer smart enough to recommend its own indexes. Proceedings of the 16th International Conference on Data Engineering, s. 101, Washington, DC, USA, 2000, IEEE Computer Society.
5. Zhang C., Yao X., Yang J.: An evolutionary approach to materialized view selection in a data warehouse environment, s. 282 – 294. IEEE Trans. Syst. 31, 2001.
6. Zilio D. C., Zuzarte C., Lightstone S., Ma W., Lohman G. M., Cochrane R. J., Pirahesh H., Colby L., Gryz J., Alton E., Liang D., Valentin G.: Recommending materialized views and indexes with IBM DB2 design advisor. 1st International Conference on Autonomic Computing, s. 180 – 188 (New York, NY, USA, 2004), IEEE Computer Society, Washington, DC, USA, 2004.
7. Sattler K., Schallehn E., and Geist I.: Autonomous Query-Driven Index Tuning. W Proceedings of the international Database Engineering and Applications Symposium (Ideas'04) - Volume 00, s. 439 – 448. IEEE Computer Society, Washington, DC, USA, 2004.
8. Schnaitter K., Abiteboul S., Milo T., Polyzotis N.: COLT: continuous on-line tuning. Proceedings of the 2006 ACM SIGMOD international Conference on Management of Data, s. 793 – 795 (Chicago, IL, USA, 2006). ACM Press, New York, NY, USA, 2006.
9. Golab L., Özsu M. T.: Issues in data stream management. SIGMOD Rec. 32, 2, s. 5 – 14, 2003.
10. Demaine E. D., López-Ortiz A., Munro J. I.: Frequency Estimation of Internet Packet Streams with Limited Space. Proceedings of the 10th Annual European Symposium on Algorithms. R. H. Möhring, R. Raman, Eds. Lecture Notes In Computer Science, tom 2461, s. 348 – 360. Springer-Verlag, Londyn, 2002.
11. Charikar M., Chen K., Farach-Colton M.: Finding Frequent Items in Data Streams. Proceedings of the 29th international Colloquium on Automata, Languages and Programming (08 – 13 lipca, 2002). P. Widmayer, F. T. Ruiz, R. Morales, M. Hennessy, S. Eidenbenz, R. Conejo, Eds. Lecture Notes In Computer Science, tom 2380, s. 693-703. Springer-Verlag, Londyn, 2002.
12. Estan C., Varghese G.: New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice. ACM Trans. Comput. Syst. 21, 3, 2003.
13. Izydorczyk J., Płonka G., Tyma G.: Teoria sygnałów. Wstęp. Helion, Gliwice, 2006.