ISBN: 978-979-17087-

ALGORITMA EFISIEN UNTUK PENGGALIAN POLA SEQUENTIAL

Budanis Dwi Meiliani
Teknik Informatika ITATS
Institut Teknologi Adhi Tama Surabaya (ITATS)
Jl. Arief Rahman Hakim no 100 Surabaya

Abstrak

Diketahui suatu basis data transaksi customer yang besar, dimana masing-masing transaksi terdiri dari customer-id, waktu transaksi, dan item yang dibeli pada transaksi itu. Dalam paper ini memperkenalkan permasalahan dalam penggalian pola sequential pada basis data tersebut. Kebanyakkan algoritma penggalian pola sequential yang ditemukan mengikuti suatu paradigma candidate maintenance dan test yaitu harus memelihara set yang telah digali sequential candidate untuk dilakukan pengujian apakah merupakan frequent sequential. Sayangnya, algoritma tersebut mempunyai skalabilitas yang lemah yaitu akan membutuhkan banyak memori dan waktu komputasi. Dalam paper ini menghadirkan suatu algoritma efisien untuk penggalian sequence tanpa pemeliharaan candidate. Algoritma ini akan menghasilkan lebih sedikit memori dan waktu komputasi pada saat penggalian pola sequential dibanding dengan algoritma yang menggunakan pemeliharaan candidate.

Kata Kunci : Penggalian Pola sequential, frequent sequential, pola sequential

Abstraction

Known bases of big transaction customer data, where each transaction consisted of the customer-id, transaction time, and item bought that transaction. In this paper introduce the problem of dig of pattern sequential the data bases. Kebanyakkan of Algorithm of dig of pattern sequential found follow a[n paradigm of candidate maintenance and testing that is have to look after to set which have been dug the sequential candidate to be [done/conducted] examination whether/what representing frequent sequential. Unhappily, the algorithm have the weak skalabilitas that is will require a lot of memory and computing time. In this paper attend a[n efficient algorithm for the dig of sequence without conservancy candidate. This algorithm will yield slimmer memory and computing time at the (time) of dig of pattern sequential compared to with the algorithm using conservancy candidate.

Keyword : Pattern sequential Pattern Mining, frequent sequential, pattern sequential.

1. Pendahuluan

Data Mining adalah suatu teknik untuk menemukan tren atau pola yang menarik dalam dataset besar untuk memandu pengambilan keputusan mengenai aktivitas yang akan datang. Kita dapat menganggap tugas data mining sebagai 'query' kompleks yang ditentukan pada level tinggi, dengan parameter yang dapat didefinisikan pengguna, dan di mana algoritma tertentu diimplementasikan. Dalam dunia nyata, data mining tidaklah hanya mengaplikasikan salah satu dari algoritma tersebut, dikarenakan data yang dihasilkan sering tidak sesuai atau tidak lengkap, kecuali hal ini dapat dipahami dan diperbaiki. Selanjutnya, analis harus bisa memutuskan jenis algoritma mining apa yang akan dipakai, menerapkannya ke subset sampel data dan variabel yang paling baik, mencerna hasil, menerapkan *decision support* dan alat mining lain, dan mengulangi proses.

Penggalian pola sekuensial (*Sequential pattern*), yang menemukan *frequent subsequences* seperti pola dalam suatu basis data yang terurut, adalah suatu masalah penting dalam data mining dengan aplikasi luas, mencakup analisa belanja pelanggan atau akses pola web, analisa peruntunan atau proses terkait dengan waktu eksperimen ilmiah,

ISBN: 978-979-17087-0-8

bencana alam dan perawatan penyakit, analisa DNA dan sebagainya.

Masalah penggalian pola sekuensial pertama kali telah diperkenalkan oleh (Agrawal, 1995) yang didasarkan studi pada urutan pelanggan, sebagai berikut: Diberikan satu basis data yang terurut, di mana masing-masing urutan terdiri dari daftar unsur-unsur dan masing-masing unsur terdiri dari satu kumpulan item, dan jika diberikan suatu ambang batas dukungan minimum, penggalian pola sekuensial berupaya untuk menemukan semua sub-urutan yang frekuensi kejadiannya dalam satuan urutan adalah tidak kurang dari *minimum support*. Jadi Masalah penggalian pola sekuensial dinyatakan "jika diberikan basis data yang terurut dan ambang batas dukungan minimum, penggalian pola sekuensial adalah untuk menemukan kumpulan yang lengkap dari pola sekuensial dalam basis data" (Han, 2005).

Kebanyakkan algoritma penggalian pola sequential yang ditemukan mengikuti suatu paradigma candidate maintenance dan test yaitu harus memelihara set yang telah digali sequential candidate untuk dilakukan pengujian apakah merupakan frequent sequential. Sayangnya, algoritma tersebut mempunyai skalabilitas yang lemah yaitu akan membutuhkan banyak memori dan waktu komputasi.

Dalam paper ini menghadirkan suatu algoritma efisien untuk penggalian sequence tanpa pemeliharaan candidate. Algoritma ini akan menghasilkan lebih sedikit memori dan waktu komputasi pada saat penggalian pola sequential dibanding dengan algoritma yang menggunakan pemeliharaan candidate. Dalam algoritma efisien ini tidak perlu menyimpan pola sequential candidate untuk dilakukan pengujian apakah memenuhi frequence sequence. Algoritma ini mendalami metode proyeksi prefix dan proyeksi basis data untuk menghasilkan pola sequential.

Dalam paper ini diuraikan menjadi 5 bagian, dimana bagian 2 menjelaskan definisi penggalian pola sequential, bagian 3 menjelaskan permasalahan pada pola sequential, bagian 4 menjelaskan algoritma efisien untuk penggalian pola sequential, dan akhir dari paper menyimpulkan studi pada bagian 5.

2. Definisi

Diberikan satu set obyek, dengan masing-masing obyek diasosiasikan dengan waktu kejadian masing-masing, maka didapatkan pola yang memprediksi ketergantungan sekuensial yang kuat antar kejadian-kejadian yang berbeda. Pola-pola sekuensial pertama, pada dasarnya, dibentuk dengan cara mencari semua kemungkinan pola yang ada. Nilai-nilai kejadian (event occurrences) dalam pola diatur berdasarkan urutan-urutan waktu kejadian.

Input data adalah sebuah set sequence yang disebut data sequences. Masing-masing data sequence adalah sebuah daftar transaksi dimana masing-masing transaksi adalah sebuah himpunan literals yang disebut items. Secara khas ada waktu transaksi yang berhubungan dengan masing-masing transaksi. Sebuah pola sekuensial juga terdiri dari sebuah daftar sets items. Permasalahannya adalah menemukan semua pola dengan sebuah user-specified min support, dimana support pola sekuensial adalah presentasi data sequences yang berisi pola.

Tabel 1. Basis data terurut dengan customer id dan transaksi time

Customer id	Transaction time	Items bought
1	1 mei 2004	C
1	5 mei 2004	D
1	10 mei 2004	A B
2	4 mei 2004	A B C
2	7 mei 2004	D E
3	5 mei 2004	A
3	8 mei 2004	C B
3	10 mei 2004	D E F
4	6 mei 2004	B C
4	9 mei 2004	D E
5	11 mei 2004	D E F

Tabel 2. Basis data customer sequence

Customer id	Customer sequence
1	{(C)(D)(AB)}
2	{(ABC)(DE)}
3	{(A)(CB)(DEF)}
4	{(BC)(DE)}
5	{(DEF)}

Dalam basis data terurut pada tabel 1, masing-masing data sequence boleh bersesuaian dengan pemilihan semua item seorang customer dan masing-masing transaksi pembelian item oleh customer dalam satu transaksi. Sebuah pola sekuensial boleh jadi 5% customer membeli A kemudian A dan B dan kemudian D dan E. Elemen / unsur-unsur sebuah pola sekuensial dapat berupa set items. Sebagai contoh pada tabel 2, untuk

ISBN: 978-979-17087-0-8

customer 1 item C diikuti dengan item D dan diikuti dengan item A dan B.

3. Permasalahan

Diberikan sebuah basis data D yang berisi customer transaksi dimana masing-masing transaksi terdiri dari field-field berikut : customer-id, transaction-time dan items yang dibeli dalam transaksi. Tidak ada customer yang mempunyai lebih dari satu transaksi dengan waktu transaksi yang sama. Jumlah items yang dibeli dalam sebuah transaksi tidak dipertimbangkan. Masing-masing item. Sebuah itemset adalah sebuah set items yang tidak kosong. Sebuah sequence adalah sebuah daftar pesanan itemsets. Support untuk sebuah sequence menggambarkan sebagai pecahan total customer sequence. Asumsi set items adalah memetakkan sebuah set dengan integer yang berdekatan. Sebuah itemset I adalah ditandai $(i_1, i_2, \ldots, i_m)$ dimana i_j adalah item. Sebuah sequence S adalah ditandai $(S_1, S_2, \ldots, S_n)$ dimana S_j adalah itemset. Sebagai contoh sequence {(A)(CB)(DEF)} dimana A, B, C, D, E, F adalah item, dan set item terdiri dari A, CB, D E F. Sebuah sequence $(a_1, a_2, \ldots, a_n)$ adalah terdapat di sequence yang lain $(b_1, b_2, \ldots, b_n)$. Jika terdapat integer $i_1 < i_2 < \ldots i_n$ seperti $a_1 \subseteq bi_1$, $a_2 \subseteq bi_2$, ..., $a_n \subseteq bi_n$. Contoh: sequence {(3)(4 5)(8)} terdapat di {(7)(3 8)(9)(4 5 6)(8)} karena (3) $\subseteq$ (3 8), (4 5) $\subseteq$ (4 5 6) dan (8) $\subseteq$ (8), dan sequence {(3)(5)} tidak terdapat di {(3 5)} dan sebaliknya.

Problem atau masalah dalam penggalian pola sekuensial adalah menemukan sequence maximal antar semua sequences yang dipunyai sebuah user-specified min_support tertentu. Masing-masing sequence maximal menghadirkan sebuah pola sekuensial. Sebuah sequence dengan batasan min_support disebut Large sequence. Dimisalkan ditentukan nilai min_support 25% maka menghasilkan dua sequence yaitu 25% dikali jumlah customer.

Sebagai contoh pada tabel 2 Basis data customer sequence menghasilkan tabel frequence 1 sequence sebagai berikut pada tabel 3:

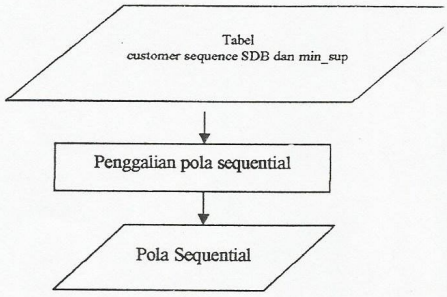
Tabel 3. Frequence 1 sequence

1 sequence	support
A	3
B	4
C	4
D	5
E	4
F	2

Pada tabel 3 yaitu tabel frequence 1 sequence menjelaskan bahwa pada item A terjadi 3 kali kemunculan pada basis data customer sequence yaitu pada customer id 1, 2 dan 3.

4. Algoritma efisien untuk penggalian pola sequential.

Dalam bagian ini dijelaskan mengenai desain algoritma penggalian pola sequential.



Gambar 1. Blok diagram Sistem Penggalian Pola Sequential

Blok diagram sistem keseluruhan penggalian pola sequential ditunjukkan pada gambar 1. Masukan dari sistem penggalian pola sequential dibatasi berupa tabel customer sequence yang terdiri dari customer id dan customer sequence, dan minimum support. Sedang hasil luaran sistem adalah pola Sequential.

```
1. FS = 0
2. F1 = scan basis data SDB untuk
   menemukan frequence 1 sequence
3. FOR (masing-masing frequence 1
   sequence F1)
4.       SDB^fl= membangun pseudo
         proyeksi basis data
5. FOR (perlakukan masing-masing
   frequence 1 sequence fl dalam F1 sebagai
   prefix)
6.       memanggil subroutine
         bideku(SDB^fl, fl, min_sup,FS)
7. return
```

Algoritma 1. Algoritma Penggalian pola Sequential

ISBN: 978-979-17087-0-8

Algoritma 1 menunjukkan algoritma penggalian pola *sequential*. Pertama pemberian nilai awal nol untuk FS atau *frequent sequential*. Kemudian scan basis data SDB sekali untuk menemukan *frequent* satu *sequences* F1. *Frequent* satu *sequences* (urutan yang sering muncul) adalah *sequences* satu item yang supportnya tidak kurang dari *minimum_suppor*. Setelah menemukan *frequent* satu *sequences* F1 kemudian membangun pseudo proyeksi basis data untuk masing-masing *frequent* satu *sequences* SDB^n.

Langkah selanjutnya perlakukan masing-masing *frequent* satu *sequences* fi dalam F1 sebagai prefix dan kemudian memanggil subroutine bideku(S_p_SDB, Sp, min_sup, FS) yang dijelaskan pada algoritma 2.

Membangun Pseudo Proyeksi Basis Data

Suatu proyeksi basis data S digambarkan sebagai $D_s = \{p|\ S \in D,\ s' = r \Diamond p$ sedemikian sehingga r menjadi prefix minimum (dari s') yang berisi s (yaitu, $s \sqsubset r$ dan $\exists r', s \sqsubset r' \sqsubset r)\}$. Pada definisi di atas, p dapat kosong.

Tabel 4 Contoh Basis Data Sequence SDB

Sequential Id	Sequence
0	Susu roti roti telur susu
1	roti telur susu telur
2	Susu roti telur susu

Sebagai contoh basis data SDB dalam Tabel 4: Proyeksi basis data untuk prefix sequence Sp = susu roti adalah :Sp_SDB = {_roti telur susu, _telur susu}. Dan proyeksi basis data untuk prefix sequence Sp = telur susu adalah :Sp_SDB = {$, _telur, $} di mana $ berarti suatu sequence dalam SDB yang berisi telur susu tetapi suffixnya adalah suatu string kosong, dan _telur berarti _telur adalah item terakhir telur susu dalam sequence tergolong itemset yang sama.

Sebagai contoh penggalian pola sequential, diketahui basis data SDB pada tabel 5, **Minimum support = 50% yaitu 50% * 6 customer = 3**

Tabel 5 Tabel basis data sequence SDB

Sequence identifier	Sequence
1	Susu roti roti telur susu
2	Roti telur susu telur
3	Susu roti telur susu
4	Roti telur telur susu roti
5	Roti susu susu roti
6	Susu roti roti

Masukan data pada algoritma Penggalian Pola Sequential adalah basis data sequence SDB dan minimum support, sedangkan luaran data adalah FS(frequent Sequential).

Tahapan algoritma Pencarian Pola Closed Sequential:

1. ***Nilai awal untuk FS = 0***
2. ***F1 = scan basis data SDB untuk menemukan frequent 1 sequence.***

   F1 (Frequence 1 sequence) adalah :
   roti = 6 support / kemunculan
   telur = 4 support / kemunculan
   susu = 6 support / kemunculan

3. ***Untuk masing-masing 1 sequence pada F1***
4. ***SDB^{f1} = membangun pseudo proyeksi basis data***

   Sp = roti:6
   Sp_SDB = _roti telur susu
   _telur susu telur
   _telur susu
   _telur telur susu roti
   _susu susu roti
   _roti

   Sp = telur:4
   Sp_SDB = _susu
   _susu telur
   _susu
   _telur susu roti

   Sp = susu:6
   Sp_SDB = _roti roti telur susu
   _telur
   _roti telur susu
   _roti
   _susu roti
   _roti roti

5. ***Untuk masing-masing frequent 1 sequential f1 dalam F1 memanggil subroutine bideku(SDB^{f1}, f1, min_sup, FS)***
6. ***return***

   Sp = roti:6
   Sp_SDB = _roti telur susu
   _telur susu telur
   _telur susu
   _telur telur susu roti
   _susu susu roti
   _roti

ISBN: 978-979-17087-0-8

Memanggil subroutine bideku(Sp_SDB, Sp, min_sup, FS)

Sp = telur:4
Sp_SDB = _susu
_susu telur
_susu
_telur susu roti

Memanggil subroutine bideku(Sp_SDB, Sp, min_sup, FS)

Sp = susu:6
Sp_SDB = _roti roti telur susu
_telur
_roti telur susu
_roti
_susu roti
_roti roti

Memanggil subroutine bideku(Sp_SDB, Sp, min_sup, FS)

Masukan algoritma subroutine bideku adalah proyeksi basis data Sp_SDB, sebuah prefix sequence Sp, minimum support. Sedangkan hasil luaran sistem adalah pola *sequential* FS.

1. FS = FS ∪ fl
2. LFI = scan proyeksi basis data Sp_SDB untuk menemukan lokal item frequent
3. FOR (untuk masing – masing i dalam LFI)
4. Sp^i = menumbuhkan Sp untuk mendapatkan prefix baru
5. SDB^{Spi} = membangun pseudo proyeksi basis data untuk prefix yang baru
6. FOR (untuk masing-masing prefix baru)
7. memanggil dirinya sendiri bideku (SDB^{Spi} , Sp^i, min_sup, FS)
8. Return

Algoritma 2
Algoritma Subroutine Bideku

Subroutine bideku(S_p_SDB, Sp, min_sup, FS) secara berulang memanggil dirinya sendiri dan bekerja sebagai berikut: Menghasilkan pola sequential FS dari prefix Sp. Untuk prefix S_p, scan proyeksi basis data Sp_SDB sekali untuk menemukan lokal frequent item LFI. Lokal frequent item adalah item sufix yang supportnya >= min_support. Menumbuhkan Sp pada masing-masing i dalam lokal frequent item LFI untuk mendapatkan prefix baru Sp^i dan membangun pseudo proyeksi database untuk prefix yang baru SDB^{Spi}. Kemudian untuk masing-masing prefix baru Sp^i, memanggil bideku (SDB^{Spi} , Sp^i, min_sup, FS)

Sebagai contoh pencarian pola Sequence dengan algoritma subroutine bideku. Diketahui basis data SDB pada tabel 4. **Minimum support = 50% yaitu 50% * 6 customer = 3.** Masukan data pada subroutine bideku adalah proyeksi basis data sequence Sp_SDB, Sp^i, dan minimum support. Sedangkan luaran data adalah FS.

Tahapan subroutine Bideku:

1. **FS = FS ∪ Sp**
2. ***Scan proyeksi basis data Sp_SDB untuk menemukan lokal item frequent (LFI).***

Untuk Sp=roti:6
FS = FS ∪ Sp = roti:6
Sp = roti:6
Sp_SDB = _roti telur susu
_telur susu telur
_telur susu
_telur telur susu roti
_susu susu roti
_roti
LFI = roti:4, telur:4, susu:5

3. ***Untuk masing-masing i dalam LFI***
4. ***menumbuhkan Sp untuk mendapatkan prefix baru (Sp^i)***
5. ***Kemudian membangun pseudo proyeksi basis data untuk prefix baru (SDB^{Spi})***

Sp^i = roti roti : 4
Sp_SDB^{spi} = _telur susu
$
$
$

Sp^i = roti telur : 4
Sp_SDB^{spi} = _susu
_susu telur
_susu
_telur susu roti

Sp^i = roti susu : 5
Sp_SDB^{spi} = $
_telur
$
_roti

ISBN: 978-979-17087-0-8

_susu roti

6. ***Untuk masing-masing perfix baru Sp^l***
7. ***Memanggil subroutine bideku(SDB^{Spi}, Sp^l, min_sup, FS)***

Untuk Sp=roti roti:4
FS = FS ∪ Sp = roti:6, roti roti:4
Sp = roti roti:4

Sp_SDB^{spi} = _telur susu
$
$
$
LFI = tidak ada

Untuk Sp = roti telur:4
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4
Sp^l = roti telur : 4
Sp_SDB^{spi} = _susu
_susu telur
_susu
_telur susu roti
LFI = susu:4

Sp^l = roti telur susu : 4
Sp_SDB^{spi} = $
_telur
$
_roti

Untuk Sp = roti telur susu:4
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4
Sp^l = roti telur susu : 4
Sp_SDB^{spi} = $
_telur
$
_roti
LFI = tidak ada

Untuk Sp = telur:4
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4
Sp = telur:4
Sp_SDB = _susu
_susu telur
_susu
_telur susu roti
LFI = susu:4

Sp^l = telur susu : 4
Sp_SDB^{spi} = $
_telur
$
_roti

Untuk Sp = telur susu:4
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4
Sp^l = telur susu : 4
Sp_SDB^{spi} = $
_telur
$
_roti
LFI = tidak ada

Untuk Sp = susu:6
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4, susu:6
Sp = susu:6
Sp_SDB = _roti roti telur susu
_telur
_roti telur susu
_roti
_susu roti
_roti roti
LFI = roti:5, telur:3, susu:3

Sp^l = susu roti : 5
Sp_SDB^{spi} = _roti telur susu
_telur susu
$
$
_roti

Sp^l = susu telur : 3
Sp_SDB^{spi} = _susu
$
_susu

Sp^l = susu susu : 3
Sp_SDB^{spi} = $
$
_roti

Untuk Sp = susu roti:5
FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4, susu:6, susu roti:5
Sp = susu roti:5
Sp_SDB^{spi} = _roti telur susu
_telur susu
$
$
_roti
LFI = tidak ada

Untuk Sp = susu telur:3

ISBN: 978-979-17087-0-8

FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4, susu:6, susu roti:5, susu telur:3

Sp = susu telur : 3

Sp_SDB^{spi} = _susu
$
_susu

LFI = tidak ada

Untuk Sp = susu susu:3

FS = FS ∪ Sp = roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4, susu:6, susu roti:5, susu telur:3, susu susu:3

Sp^i = susu susu : 3

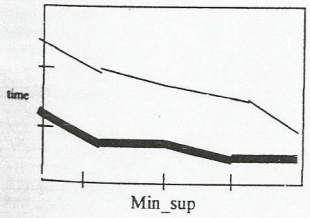
Sp_SDB^{spi} = $
$
_roti

LFI = tidak ada

Hasil keseluruhan pola sekuential yang dihasilkan adalah roti:6, roti roti:4, roti telur:4, roti telur susu:4, telur:4, telur susu:4, susu:6, susu roti:5, susu telur:3, susu susu:3

5. Evaluasi Performance

Gambar 2 menunjukkan waktu pelaksanaan relatif dari algoritma efisien dan algoritma yang menggunakan candidate sequence berdasarkan minimum supportnya. Sekalipun algoritma yang menggunakan candidate sequence menghabiskan banyak memori, biaya menemukan support untuk banyak candidate telah dipastikan akan membutuhan banyak waktu pencarian lebih besar.



Gambar 2. Waktu Pelaksanaan

Pada gambar 2 digambarkan waktu komputasi yang dihasilkan pada algoritma efisien dan algoritma yang menggunakan candidate sequence. Garis tebal pada tabel adalah garis yang dihasilkan dengan menggunakan algoritma efisien penggalian pola sequential.

6. Simpulan

Penggalian pola sekuensial (*Sequential pattern*), yang menemukan *frequent subsequences* seperti pola dalam suatu basis data yang terurut, adalah suatu masalah penting dalam data mining dengan aplikasi luas, mencakup analisa belanja pelanggan atau akses pola web, analisa peruntunan atau proses terkait dengan waktu eksperimen ilmiah, bencana alam dan perawatan penyakit, analisa DNA dan sebagainya. Dalam penggalian pola sequential mempunyai kesamaam pada penggalian itemset frequent, dan perbedaan dalam pendekatan pencarian pola sequential yaitu dengan menggunakan candidate sequence dan tanpa menggunakan candidate sequence.

7. Daftar Pustaka

Jianyong Wang and Jiawei Han, 2004, *BIDE: Efficient Mining of Frequent Closed Sequences*, Departement of Computer Science University of illiosi at Urbana-Champaign, Illinois, U.S.A.

R. Agrawal and R. Srikant, 1995, *Mining Sequential Pattern*, IBM Almaden Research Center 650 Harry Road, San Jose, CA 95120