PENEMUAN POLA AKSES PENGGUNA WEB MENGGUNAKAN ALGORITMA FREQUENT WEB DOCUMENT PATTERN (FWDP) DAN AZAS APRIORI STUDI KASUS PADA LOG WEBSERVER UNIVERSITAS RESPATI YOGYAKARTA

Indra Listiawan[1)]

[1)] *Manajemen Informatika Universitas Respati Yogyakarta*
Jl. Adisucipto km 6.3 ,Depok, Sleman, Yogyakarta
Email : indra.unriyo@yahoo.com[1)]

Abstrak

Website sebuah lembaga merupakan profile lembaga tersebut bagi masyarakat yang berhubungan langsung maupun tidak langsung terhadap lembaga. Beberapa permasalahan terkait kinerja sebuah web antara lain adalah kecepatan dan ketepatan penyajian informasi yang dibutuhkan masyarakat.

Penemuan pola akses pengguna website adalah salah satu solusi untuk mempersiapkan halaman-halaman web sesuai dengan pola akses. Pola akses web dapat diimplementasikan sebagai link-link pada halaman-halaman web tersebut. Adanya link-link yang bersesuaian dengan akses pengguna akan mempercepat akses pengguna menuju halaman web yang diinginkan.

Proses penelitian dilakukan dalam beberapa tahap. Pertama Preprosessing data dilanjutkan dengan proses pembentukan tree dengan menerapkan algoritma *FWDP tree* untuk menemukan halaman web yang memenuhi frequent item web minimal. Bagian terakhir adalah penerapan algoritma Association Rule sesuai dengan azas apriori untuk mendapatkan pola akses pengguna.

Kata kunci: *Pola Akses, Data Log, Algoritma FWDP, Rekomendasi Link.*

1. Pendahuluan

Penelitian untuk pengembangan web yang menggunakan pola adalah aplikasi *WebWatcher* [1] yang menggunakan model pengguna untuk memperkirakan *link* apa yang akan diikuti pengguna dari suatu halaman web. Sebelum memasuki *website* , pengguna ditanya tentang apa yang mereka inginkan. Jawaban pengguna dan urutan halaman yang dikunjungi, digunakan untuk memperbaiki model yang dirumuskan.

Penelitian lain yang pernah dilakukan adalah penelitian yang mengembangkan metode *pattern-growth* yang efisien dan baru untuk data mining berbagai pola frekuen dari database yang besar [2]. Metode *Patern-Growth* diadopsi melalui pendekatan *divide-and-conquer* untuk mendekomposisi database dan pekerjaan-pekerjaan datamining. Kemudian digunakan metode pertumbuhan fragmen yang berpola untuk menghindari proses tes dan pembangkitan kandidat yang memakan biaya. Lebih dari itu, data struktur yang efektif diusulkan untuk menekankan informasi yang krusial tentang pola frekuen dan menghindari biaya tinggi seperti perulangan pemindaian database. Studi menunjukkan bahwa metode *pattern –growth*, *FP-Growth* dan *H-mine* efisien dan mempunyai skalabilitas. Walaupun demikian masih terdapat problem pada banyak proses rekursi yang dilakukan pada database sehingga membutuhkan lebih banyak tempat penyimpanan.

Algoritma Frequent *Web Document Pattern(FWDP)* diusulkan untuk mendapatkan pola akses pengguna [3]. Algoritma ini mengubah data mentah yaitu web log menjadi struktur data yang lebih sesuai yang kemudian dengan menggunakan algoritma FWDP-mine dilakukan penambangan data terhadap struktur data yang berbentuk FWDPTree. Kelebihan algoritma ini adalah tidak membaca database berulang-ulang dan tidak membangkitkan kandidat, sebelum dapat menambang panjang maksimum dari pola. Pola frequent yang ditambang adalah satu set dokumen web yang saling terkait dan mejadi daftar halaman yang direkomendasikan.

2. Pembahasan

2.1. Deskripsi Sistem

Proses penemuan pola akses pengguna terdiri atas tiga proses yaitu proses preprosesing data, proses pembentukan *FWDP tree*, proses data mining. Preprosesing data adalah proses mengambil data dari file log web server sampai persiapan struktur data untuk proses *FWDP tree*. Proses data mining dilakukan terhadap struktur *FWDP tree*, sehingga dihasilkan pola akses pengguna.

Sistem akan melakukan pembersihan dan pemilihan data dari file log. Sistem akan melakukan proses pembentukan *FWD tree* dan proses terakhir adalah data mining. Proses pembentukan *FWDP tree* berdasarkan frekuensi halaman web yang diakses pengguna dan nilai support minimal. FWDP tree dibentuk melalui algoritma *FWDP tree*. Proses data mining dilakukan menggunakan azas Apriori yang menghasilkan pola akses pengguna pada waktu tertentu.

2.2. Algoritma Frequent Web Document Pattern

Algoritma *Frequent Document Web Pattern* terdiri atas dua algoritma yaitu algoritma untuk

membuat struktur tree (FWDP tree) dan algoritma untuk melakukan mining terhadap *FWDP tree* (*FWDP mine)* [3]. Gambar 1 memperlihatkan algoritma *FWDP tree*

input : Table transaksi Trans dan batas support minimum SUPmin
Output: FWDP-tree
langkah 1 : baca database sekali, ambil F yaitu himpunan dokumen yang sering muncul. Urutkan F sesuai support secara dari besar kekecil sebagai Flist. Buat Arraynya (Farray)
langkah 2: Buatlah header H sebagai penyimpan
H terdiri atas H name, H first dan H last. H first adalah node link pertama dan H last node link terakhir. Isi Hfirst sesuai dengan isi Flist.Tetapkan ROOT misal T
langkah 3 Untuk setiap transaksi pada trans
masukkan dokumen yang sering muncul sesuai batas support minimum dalam P sehingga [p|P] dimana p elemen dari P.
jalankan fungsi insert_tree ntuk setiap anggota P
For each p do {
insert_tree(p,T)
}
}
Function insert_tree(p,T) {
bila T punya child N sehingga
N.nama_dokumen=p.nama_dokmen
maka tambah count dengan 1
bila tidak { buat node baru newNode.
masukkan newNode sebagai kanan terakhir anak dari root.
setelah itu maka newNode menjadi anak sebelah kiri dari parent.
beri nilai count=1
tetapkan Child-link=null
tetapkan parent-link=parent node}

Gambar 1.*Algoritma FWDP*

2.3. Preprocessing Data

Analisa data

Data awal berupa file log yang tersimpan di server Universitas Respati Yogyakarta yang diambil sebagai sampel. Gambar 2. menampilkan contoh file log web server setelah dibersihkan. Susunan datanya adalah : clientAddress, Identity, authuser, Time, Request, status, bytes, referer, User-Agent. Tabel 1. berisi keterangan dari masing-masing bagian data file log web server.

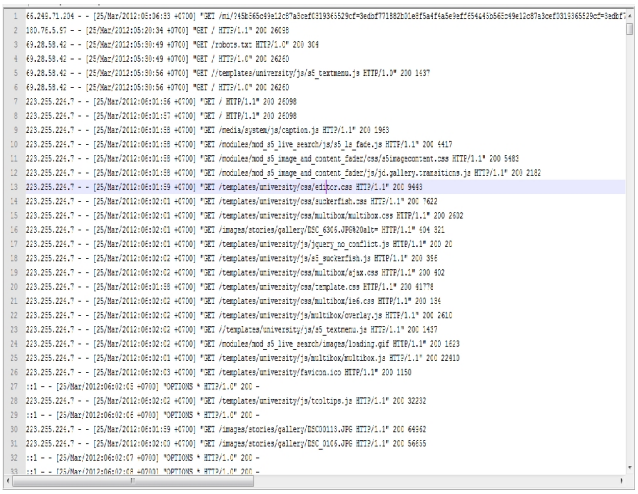
Tabel 1. Keterangan komponen file log web server

IP address	adalah alamat Ip pengguna
Identity	adalah informasi identitas pengguna.
Authuser	adalah fasilitas yang digunakan pada saat fasilitas SSL (Security Socket Layer) diaktifkan. Pengguna bisa menggunakan fasilitas ini untuk mengirim atau menerima informasi yang bersifat rahasia.
Time	adalah tanggal dan waktu pada saat web di minta oleh web browser ke internet.
Request	adalah objek yang diminta oleh browser.
Status	adalah nilai integer yang menunjukkan status permintaan .
Bytes	adalah jumlah bytes yang dikembalikan dalam proses permntaan.
Referrer	adalah string text yang dikirim oleh pengguna yang menunjukkan sumber resmi dari permintaan atau link.
User-Agent	adalah nama dan versi dari software pengguna yang digunakan untuk mengirimkan permintaan

Data ip address dan waktu digunakan untuk menentukan sesi dengan batasan 1 sesi 30 menit [4].

Komposisi data yang akan diambil dari filelog tersebut adalah data Ipaddress, httpRequest dan waktu . Oleh karena data pada httpRequest belum bersih dari data yang dibutuhkan, masih terdapat ekstensi file gambar (jpg dan pnp), rss, pdf, txt maka harus dilakukan pembersihan terlebih dahulu. Pembersihan ini perlu dilakukan agar file log yang mengandung ekstensi file yang bukan merupakan halaman web tidak terikut dalam proses pemotongan dan penyimpanan dalam table dtlog.

Proses pertama adalah pembersihan dari file log yang mencatat akses file–file tertentu yang tidak dibutuhkan seperti : jpg, bmp, pdf, txt, css. Isi file yang telah dibersihkan dapat dilihat pada pada Gambar 2. sudah tidak tampak data dengan ekstensi .txt maupun .css Pembersihan dan pemotongan isi file log web server menggunakan ekspresi regular dari Perl. Data yang telah dibersihkan disimpan didalam tabel dtlog.

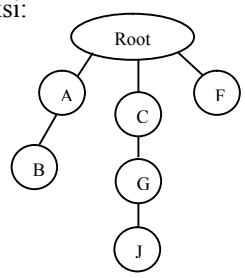


Gambar 2. *Contoh isi file log web server*

Tahapan pengelompokan transaksi data berdasarkan sesi dan Remove Backtrack

File yang telah melalui proses pembersihan kemudian dikelompokkan berdasarkan transaksi dan sesi. Sesi menentukan frekuensi akses sebuah halaman web. Satu sesi (transaksi) dibatasi maksimal 30 menit. Bila pengguna mengakses dokumen web dengan alamat yang sama lebih dari 30 menit maka terekam data 2 transaksi dengan alamat url yang sama.

Penggunaan class Preprocessor dilakukan untuk membaca data dari tabel dtlog lalu menyimpan data-data tertentu dalam tabel tranurl,tranip dan url juga mengelompokkan data sesuai dengan sesi (waktu transaksi) yang sama. Data yang telah dikelompokkan akan menjalani proses remove backtrack untuk mendapatkan url yang berarti. Proses remove backtrack menyerupai proses palindrom hanya saja untuk beberapa url yang sama dan berurutan akan direduksi menjadi satu url. Proses ini dapat diperlihatkan seperti Gambar 4. yang menampilkan tree akses halaman web dan terjadi dalam satu transaksi:



Gambar 3. *Tree akses halaman web*

Tabel 2. Transaksi data yang mengalami proses *remove backtrack*.

ID Transaksi	Data Transaksi Awal	Data Transaksi setelah Proses
100	ABACGJGCF	BJF
200	ABEBADH	EDH
300	ACGJGI	ACJI
400	HDABACCCF	HDBCF
500	EBACGHGI	EBACHI
600	ACFCGI	AFGI
700	DABEBACF	DECF
800	ABEBACF	ECF
900	ADHDACGJ	HCGJ

A, B, C, G, J, F adalah url, sedangkan *Root* adalah /*home* dari sebuah web. Pada path pertama yaitu root-*A-B-A*-root akan dihasilkan url *B*, pada path kedua yaitu root-*C-G-J-G-C*-root akan dihasilkan url *J* demikian pula pada path yang ketiga akan dihasilkan : *F*, sehingga pada satu sesi ini akan dihasilkan untaian url *B-J-F*. Proses *remove backtrack* dilakukan dengan menggunakan *class PalindromTree* dan *Preprocessor*. Proses berikutnya adalah menyimpan data dalam di tabel *burl*. Tabel 2 menunjukkan transaksi data yang mengalami proses *remove backtrack*.

Data yang telah disimpan didalam tabel *burl* dipilih berdasarkan support minimal yang ditentukan yaitu 3 dan diurutkan *descending* berdasarkan frekuensi diakses tiap *url*. Tabel 3. memberikan gambaran hasil pengurutan dan filter data:

Tabel. 3. Tabel hasil pengurutan berdasarkan frekuensi dan penyaringan berdasarkan suport

ID Transaksi	Urutan Transaksi Awal	Hasil Penyaringan dan Urutan Transaksi sesuai frekuensi
100	BJF	FB
200	EDH	EHD
300	ACJI	CAI
400	HDBCF	CFHBD
500	EBACHI	CEHABI
600	AFGI	FAI
700	DECF	CFED
800	ECF	CFED
900	HCGJ	CH

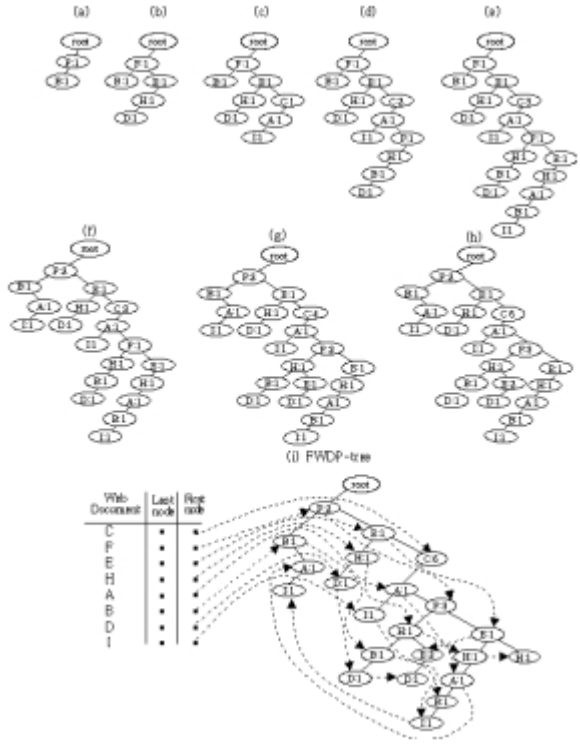
Data yang telah dikelompokkan sesuai sesinya kemudian dipisah-pisahkan sesuai data yang diperlukan seperti data *Ipaddress*, halaman *web* dan *waktu*. Kemudian data-data tersebut disimpan dalam tabel.

2.2. Pembuatan *FWDPTree*

FWDPTree membentuk *tree* yang terdiri dari *node*, *parent*, *child, siblink* dan link untuk header. Header berisi susunan perwakilan tiap *node* dari masing-masing path transaksi. Setiap node dalam *FWDTree* telah memenuhi support minimal. Bentuk *tree* ini dibuat sedemikian rupa sehingga data transaksi yang memiliki kesamaan awalan(prefiks) tidak akan ditulis ulang melainkan dinaikkan perhitungan jumlah frekuensi (jumlah diakses).

Proses pembentukan *tree* adalah sebagai berikut. Setelah proses prepocessing dengan penyimpanan data terakhir pada tabel *Burl* kemudian tabel *Burl* dibaca pertransaksi dan disimpan dalam *Arraylist Nodes*. Pembentukan *tree* dilakukan dengan pembacaan pada data *ArrayList Nodes* per transaksi sebagai *node* satu persatu sesuai dengan urutan transaksi (*tranid*) dan frekuensi (dilakukan pengurutan berdasarkan frekuensi diakses). Pembentukan *tree* dimulai dengan menentukan *node* sebagai anak (*child*) dari *ROOT* yang kemudian ditetapkan sebagai *current node*.

Pada transaksi berikutnya dibentuk *siblink node* yaitu *node* yang setingkat dengan *child node* apabila transaksi yang dibaca berbeda dan belum terdapat *siblink*, proses dilanjutkan dengan penentuan *curent* pada *node* yang baru di bentuk. Pada pembacaan data berikutnya apabila node yang dibaca memiliki transaksi yang sama dengan *current* maka akan dijadikan *child* dari current *node*, bila data yang dibaca memiliki transaksi dan *url* yang sama dengan *current node* maka frekuensi *current* akan bertambah.



Gambar 4.*FWDP Tree yang dihasilkan*

Pembentukan *tree* dibarengi dengan membuat *link* antar *node* yang sama yang akan digunakan sebagai panduan pengambilan *node* yang berasosiasi dalam proses *data mining*. Proses *data mining* menggunakan *htable* yang bertype *hash map* untuk menampung *link* antar *node* yang sama dari *node* pertama sampai terakhir.

2.3.. Proses data mining.

Proses *data mining* dilakukan setelah *FWDP tree* terbentuk. Proses data mining menggunakan *htable* bertype hashmap untuk menampung *header node* yang menjadi panduan proses data mining. Proses mining membaca *path* dari untai data di dalam struktur *FWDP tree* Proses akan melakukan pembacaan *node-node berdasarkan header node yang terdapat dalam htable. Header node* yang telah dibaca tidak akan dibaca kembali. Proses *data mining* ini akan dilakukan mulai pada posisi *pointer (current node)* sampai root. Proses ini dilakukan sampai seluruh *node* terbaca, sehingga akan diperoleh keterkaitan jumlah kemunculan antar satu node dengan *node* yang lainnya.

Proses mining dengan menerapkan prinsip apriori digunakan untuk mendapatkan pola akses. Proses mining dilakukan terhadap struktur FWDP tree. Pola akses adalah halaman-halaman web yang saling terkait. Gambar 5. memperlihatkan algoritma FWDP mine.

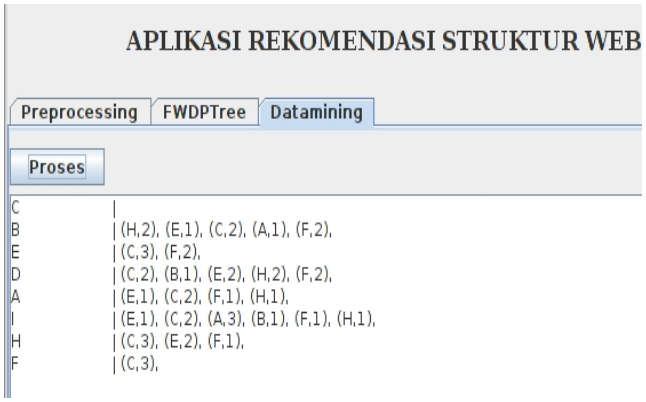
```
procedure FWDP-mine() {
   for each ai do {  /*Flist =a1,a2...*/
          Pattern P={}  /*dimana
P={(x1,s1),(x2,s2)...(xn,sn)}, x nama
                                node dan s
                    supportnya*/
                    for all the path of ai do {
                       sup(ai) support ai pada path
                     ptr = parent node  dari ai
                              for all each node from ptr
to root do {
          if (ptr = ptr.parent link.child-link)
             then { create pattern node(xptr,sptr)
             if (node(xptr,sptr) = node(x,s) in P)
            then s += sptr
             else insert node(xptr,sptr) in P
```

Gambar 5.*Algoritma FWDP mining*

Proses *data mining* membaca data pada htable yang berupa link beserta node. Proses ini memasukkan *node* yang terkait dengan *node link* kedalam *hash map P*. Proses minng akan menyimpan node parent dalam P. Sebelum penyimpanan dilakukan akan dicek apakah parent dari current node (pointer) memiliki anak (child) dan child tersebut urlya sama dengan current node bila memenuhi maka parent dismpan di P, tetapi bila bukan child tetapi siblink, maka parent akan menjadi current node (pointer) dan pengecekan dilakukan kembali untuk mengambil node parent. Bila ditemukan url parent ada yang sama dengan url yang sudah disimpan dalam P maka frekuensi dari url tersebut ditambah. Proses ini akan berjalan terus hingga pointer sampai di root.

2.4 Hasil Penelitian

Pola akses dihasikan melalui proses datamining menggunakan data sembarang sebagai simulasi yang diwakili abjad menghasilkan pola seperti pada gambar 6



Gambar 6 . Pola akses dengan data abjad

Melalui jendela aplikasi terlihat keterkaitan antar halaman web yang merupakan pola akses pengguna. Pembacaannya adalah sebagai berikut pengguna yang mengakses halaman B cenderung mengakses halaman H sebanyak 2 kali, sedangkan pengguna halaman E cenderung akan mengakses halaman C dengan frekuensi sebanyak tiga kali. Oleh karena ada syarat batasan jumlah minimal halaman diakses agar dapat diekstrak sebagai pola akses pengguna yaitu minimal 3 kali diakses maka halaman-halaman yang jumlah frekuensi aksesnya kurang dari 3 tidak akan diekstrak sebagai pola akses pengguna sehingga hasil akhirnya adalah seperti tampak pada tabel 4. pola akses pengguna.

Tabel 4. Pola akses pengguna

Dokumen	Urlid	Pola	Urlid
E	7	C	3
I	10	A	1
H	9	C	3
F	6	C	3

sedangkan pola akses yang dihasilkan proses data mining menggunakan data sesungguhnya dari webserver Universitas Respati Yogyakara adalah seperti yang ditunjukkan pada gambar 7 memperlihatkan berupa keterkaitan halaman

Gambar7. Pola akses pengguna yang diperoleh

3. Kesimpulan

Setelah menyelesaikan penelitian ini , dapat diambil beberapa kesimpulan sebagai berikut:

1. Penerapan algoritma FWDP terhadap file log menghasilkan pola akses. Implementasi pola akses pengguna menghasilkan web dengan fasilitas rekomendasi link-link web yang sering diakses pengguna. Link-link web akan berubah bilamana pola akses pengguna berubah.
2. Pola akses pengguna yang dihasilkan tidak banyak mengalami perubahan mulai 76 jam sampai 100 jam. Perbedan yang terjadi sebesar 1 link. Perbedaan lebih dari 1 link terjadi saat pola dicari dari data log kurang dari 76 jam dan lebih dari 100 jam.
3. Sistem belum seluruhnya berjalan secara otomatis (offline) sehingga masih ada sesi yang tidak ikut diproses.

Saran untuk penelitian ini adalah agar pengguna sering mengakses halaman web maka isi halaman web harus terkait dengan kebutuhan pengguna, misalnya halaman web e-learning yang berisi hal-hal yang terkait dengan perkuliahan.

Daftar Pustaka

[1] Freitag, D., Joachims, T., Mitchell, T., 1997, WebWatcher: A Tour Guide for the World Wide Web, IJCA'97
[2] Pei, J., 2002, *Pattern-Growth Methods For Frequent Pattern Mining*, A Thesis, Computing Science, Simon Fraser University
[3] Jun,W., Lee, J., 2008, *Adaptive School Web Site Construction Algorithm Using Association Rules*, IJCSNS International Journal of Computer Science and Network Security, Vol.8 No.1
[4] Palade, V., Velásquez, J, 2008, Adaptive *Website: A knowledge Extraction From Web Data Approach*, IOS Press, Amsterdam

Biodata Penulis

Indra Listiawan, memperoleh gelar *Master Computer Science* (M.Cs), Program Pasca Sarjana Ilmu Komputer Fakultas MIPA Universitas Gajah Mada Yogyakarta, lulus tahun 2012.Saat ini menjadi Dosen di Universitas Respati Yogyakarta.