MAKARA, SAINS, VOL. 8, NO. 2, AGUSTUS 2004: 76-84

QUERI GANDA PADA SISTEM TEMU-KEMBALI INFORMASI BERBASIS JARINGAN INFERENSI

Yahma Wisnani

Departemen Matematika, FMIPA, Universitas Indonesia, Depok 16424, Indonesia

E-mail : ywisnani@yahoo.com

Abstrak

Queri ganda adalah sebuah queri yang mengkombinasikan queri *Boolean* dan probabilistik pada sistem temu-kembali informasi berbasis Jaringan Inferensi, sistem tersebut terdiri dari dua komponen yaitu jaringan dokumen dan jaringan queri, kedua jaringan dihubungkan oleh busur antara istilah dokumen dan istilah queri. Jaringan dokumen membangun sebuah berkas pembalikan dokumen dan jaringan queri dievaluasi dengan menggunakan matrik kanonik. Proses penyesuaian antara istilah queri dan istilah dokumen menghasilkan sekumpulan dokumen terambil yang relevan. Hasil percobaan menunjukkan formulasi queri ganda secara siknifikan meningkatkan kinerja sistem temu-kembali jika dibandingkan dengan queri *Boolean* atau probabilistik.

Abstract

The Multiple Query of Information Retrieval System Based on Inference Network. The multiple query is a query that combines *Boolean* and probabilistic query on the information retrieval inference network system, the system consists of two components i.e. a dokument and a query network, they are joined by links between the representation and query concepts. The document network build an inverted belief list and the query network are evaluated by using canonical matrix. The similarity process between representation and query conceps yields a set of relevant document retrieved. The experiment has showed that the use of multiple query formulations will significantly improve the retrieval perfomance, compared to either the *Boolean or* probabilistic query.

Keywords: information retrieval, inverted list, link matrix, canonical matrix, precision-recall table.

1. Pendahuluan

Toffler [1] menulis dalam bukunya bahwa faktor kunci pada abad ke-21 adalah "informasi", dimana kekuatan, kekuasaan, kekayaan dan pengaruh bertumpu pada penguasaan informasi. Arus gelombang informasi mengalir demikian deras, disertai implikasi perubahan teknologi yang amat cepat. Di sisi lain manusia membutuhkan pengetahuan yang luas, yang tak terbatasi oleh informasi material, sains empiris dan teknologi praktis. Oleh karenanya dapat dipastikan bahwa hanya individu atau kelompok yang menguasai informasi yang dapat meraih kesuksesan.

Informasi yang tersimpan pada suatu koleksi dokumen dapat diakses melalui komputer jika komputer tersebut sudah dilengkapi dengan *Information Retrieval System* (Sistem Temu-kembali Informasi). Sistem temu-kembali informasi berhubungan dengan pemilihan suatu objek dari sebuah koleksi dokumen; objek tersebut merupakan informasi yang mungkin diminati oleh pemakai, dapat berbentuk dokumen teks, benda dalam museum atau sembarang bentuk yang dikumpulkan yang kelak mungkin diperlukan [2].

Beberapa metode sistem temu-kembali informasi antara lain metode: *Boolean* [3], Probabilistik [4], Ruang Vektor [5], Fuzzy [6], P-Norm [7], Jaringan Inferensi [8]. *Information need* (permintaan informasi) pemakai disampaikan dengan cara memformulasikan queri, namun queri untuk metode *Boolean* dan *P-norm* harus diformulasikan secara lengkap pada

saat pemakai memformulasikan permintaan informasinya. Akibatnya sistem seringkali tampak kurang efisien [9] yang disebabkan oleh individu yang berbeda akan memakai istilah yang berbeda untuk domain subjek yang sama, sebaliknya istilah yang sama seringkali mempunyai makna ganda [10]. Oleh karenanya permintaan informasi pemakai adalah sangat individual dan internal sifatnya yang kadang tidak diketahui secara tepat dan lengkap [11]. Sistem temu-kembali dengan Jaringan Inferensi dirancang agar pemakai dapat memformulasikan permintaan informasinya secara bertahap, artinya setelah queri pertama diproses lalu pemakai membaca dan mengidentifikasi istilah penting dari dokumen yang terambil, bila dinilai kurang sesuai maka pemakai dapat menyempurnakan permintaannya dengan cara mencantumkan istilah penting tersebut dalam queri berikutnya (secara otomatis sistem akan memperhalus struktur jaringan queri yang ada). Hal ini menunjukkan adanya proses penyesuaian istilah queri dengan istilah yang ada dalam dokumen [12].

Ide ini sejalan dengan umpan balik pemakai yang mengajukan queri baru dengan menambahkan beberapa istilah baru yang dipilih setelah membaca dokumen yang terambil dari queri sebelumnya, yang terbukti dapat meningkatkan kinerja sistem [13].

Disamping itu sistem yang dikembangkan dalam penelitian ini memberi kebebasan kepada pemakai untuk memformulasikan permintaan informasinya baik dengan queri *Boolean* (queri menggunakan operator *and*, *or*, atau *not*), dengan pendekatan probabilistik (queri menggunakan operator *sum* dan *wtd*) ataupun campuran dari kedua tipe tersebut yang formulasinya dinamakan sebagai queri ganda. Ide mengkombinasikan queri *Boolean* dan probabilistik dipandang sebagai mengkombinasikan *multiple sources of evidence*. Hal ini dimaksudkan agar menduga relevansi antara dokumen dan queri lebih logis sehingga dapat meningkatkan kinerja sistem [14]. Teori tentang bagaimana mengkombinasikan berbagai *sources of evidence* ini dilandasi oleh teori Jaringan Inferensi Bayes [15].

Studi ini membahas tentang sistem temu-kembali informasi berbasis jaringan inferensi, memeriksa besaran-besaran yang sesuai dengan fungsi pembobotan, membahas metode peringkat dokumen untuk 5 macam operator untuk formulasi queri, mengajukan berbagai formulasi queri, baik queri *Boolean*, dengan pendekatan probabilistik maupun queri ganda serta memeriksa kinerja sistem dengan tabel *Recall-Precision*.

Pada umumnya terdapat 3 masalah pada sistem temu-kembali yaitu:
1. Bagaimana memberi bobot istilah-istilah sebagai penciri setiap dokumen dalam koleksi dokumen.
2. Teknik apa yang digunakan untuk memperingkat dokumen terambil agar dokumen relevan yang terambil berada disebelah atas, sebagai respon terhadap queri.
3. Bagaimana mengukur kinerja sistem temu-kembali yang dikembangkan? Bagian ini bukan merupakan bagian dari sistem temu-kembali, tetapi merupakan alat ukur sistem.

Masalah pertama diatasi dengan pembahasan pada jaringan dokumen, masalah kedua dengan pembahasan pada jaringan queri, sedangkan masalah ketiga dengan menggunakan tabel *Recall-Precision*. Masalah pertama dan kedua merupakan faktor utama yang mempengaruhi kinerja sistem. Sistem yang baik akan meletakkan dokumen relevan diurutan sebelah atas sehingga pemakai mudah mengidentifikasi dokumen terambil yang relevan dengan keinginannya.

2. Metode Penelitian

Sistem temu-kembali informasi berbasis Jaringan Inferensi direpresentasikan dalam suatu *directed, acyclic dependency graph* yang terdiri dari 2 komponen yaitu jaringan dokumen dan jaringan queri. Jaringan dokumen menghasilkan bobot istilah dalam dokumen yang menghasilkan berkas pembalikan dokumen, jadi menjawab masalah pertama. Sedangkan jaringan queri menghasilkan matrik *link* untuk 5 operator yang dapat digunakan oleh pemakai yaitu operator *and, or, not, sum* dan *wtd*, bentuk umum dari kelima matrik *link* tersebut menghasilkan matrik kanonik. Untuk menghemat *time* dan *space* queri dievaluasi menggunakan matrik kanonik. Evaluasi queri menghasilkan dokumen terambil yang diperingkat berdasarkan perhitungan antara bobot istilah dengan matrik kanonik. Proses perhitungan ini menjawab masalah kedua.

Jaringan dokumen terdiri dari kumpulan dokumen yang menggunakan beragam skema yang merepresentasikan dokumen. Jaringan ini menyimpan semua informasi yang ada dalam koleksi dokumen yaitu data tentang istilah dokumen serta bobot istilah tersebut dalam dokumen. Semua istilah dokumen disimpan dalam berkas pembalikan dokumen. Berkas pembalikan dokumen dibentuk hanya sekali untuk suatu koleksi dokumen dimana nilainya tidak berubah selama dan setelah pemrosesan queri. Berkas pembalikan dokumen dihasilkan setelah melalui beberapa proses yaitu:
- *Indexing,* adalah proses pemilihan istilah yang akan digunakan sebagai penciri dokumen dengan tahapan sbb:
- *Parsing* dokumen, proses memilih kata-kata yang akan digunakan sebagai istilah dokumen.

- *Stemming*, proses memenggal imbuhan kata untuk mendapatkan kata dasar benar, misal kata "pengintegralan", "integralkan", "diintegralkan", "mengintegralkan" menjadi kata "integral".
- *Stoplist*, proses membuang kata buangan (yang, untuk, dari, ke, pada, jika, maka, dan, di, …. dll).

Pembobotan istilah dokumen, merupakan bagian dari pada proses *indexing*. Pada bagian ini dilakukan penghitungan peluang setiap istilah pada dokumen bernilai *true* dengan rumus sbb:

$P(t_i|d_j=true)=\alpha+(1-\alpha)\times ntf_i \times nidf_{io}$ dan

$P(t_i|d_j=false)=\beta, \quad \beta \in [0,\ 0.5)$

dimana kepercayaan terhadap suatu istilah dalam dokumen dipengaruhi oleh kemunculan istilah dalam dokumen (*tf*) dan frekuensi kemunculan istilah tersebut dalam koleksi dokumen (*idf*). Komponen frekuensi kemunculan istilah dalam koleksi dokumen diekspresikan sebagai:

dimana n sebagai jumlah dokumen dalam koleksi dan df_i sebagai jumlah dokumen dalam koleksi yang mengandung istilah i. Normalisasi dari tf_{ij} ($=ntf_{ij}$) adalah perbandingan antara frekuensi istilah i dalam dokumen j dengan maksimum frekuensi suatu istilah dalam dokumen j, sedangkan $nidf_i$ normalisasi dari idf_i

Pembahasan jaringan queri dibagi dalam 2 tahap, tahap pertama merupakan landasan teori tentang matrik *link* yang menghasilkan matrik kanonik. Tahap kedua mengenai algoritma untuk implementasi.

Jaringan queri terdiri dari node tunggal yang merepresentasikan permintaan informasi pemakai dan satu atau beberapa node queri direpresentasikan sebagai ekspresi permintaan informasi oleh pemakai. Pemakai dapat mengekspresikan permintaannya dengan mengajukan queri. Queri dapat diformulasikan dengan memgunakan operator *and, or, not, sum,* dan *wtd*. Operator *and*, *or* maupun *not* mengadopsi logika proposisi, sedangkan operator *sum* dan *wtd* merupakan perluasan ide dari logika proposisi. Logika proposisi tersebut direpresentasikan dalam bentuk matrik *link*, jadi untuk 5 operator disediakan 5 matrik *link*, setiap operator berkorespondensi dengan matrik *link* yang sesuai. Kelima matrik *link* menghasilkan matrik kanonik yang akan digunakan dalam mengevaluasi queri.

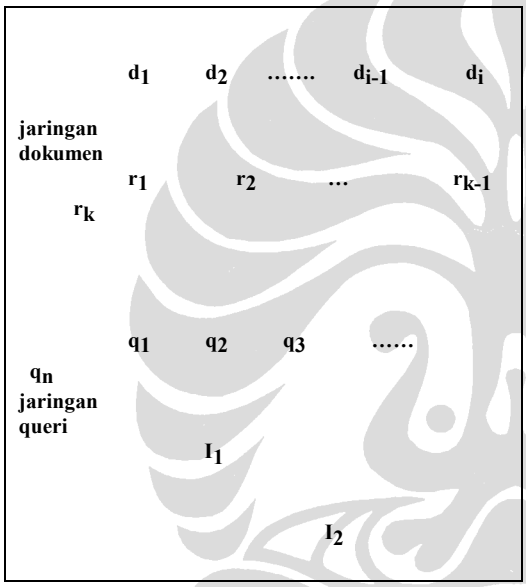
Jaringan dokumen dan queri digabungkan oleh busur berarah antara istilah dokumen dan istilah queri semua node dalam jaringan bernilai *true* atau *false*. Node bernilai *true* artinya pada saat itu node tersebut adalah satu-satunya node yang sedang aktif atau yang diinstanstiasi sementara node lainnya bernilai *false*. Pada satu saat hanya satu dokumen yang diinstantiasi. Jaringan dokumen terdiri dari kumpulan dokumen d_i dan representasi istilah dokumen r_i sedangkan jaringan queri terdiri dari queri q_i, i=1,2,….n dan sebuah permintaan informasi I. Jika pada Gambar 1 node I_2 dan busur yang mengarah kedirinya ditiadakan maka jaringan disebut model dasar untuk jaringan Inferensi, sedangkan Gambar 1 menunjukkan skema untuk queri ganda.

Untuk semua node bukan akar dalam Jaringan Inferensi menurut teori Jaringan Inferensi Bayes dapat diduga peluang suatu node menerima sehimpunan nilai dari node *parent*-nya. Jika node *a* mempunyai himpunan *parent* $\pi_a=\{p_1$, ...,

$p_n\}$, maka dapat diduga nilai $P(a|\, p_1, ..., p_n)$.

Untuk menentukan dugaan pada suatu node yang merupakan node bukan akar diperlukan matrik *link* L[i,j], dimana $i \in \{0,1\}$, $0 \le j < 2^n$. Artinya proposisi *a* bernilai biner yaitu *a true* atau *false*. Matrik berukuran 2 x 2^n diperlukan untuk merepresentasikan *n parent* dari *a*, yang menspesifikasi peluang untuk semua kombinasi nilai *parent*. Kombinasi yang mungkin dari *parent* mengkondisi nilai matrik *link* untuk menyediakan informasi *diagnostic* dengan memperhitungkan komponen *predictive* dari sekumpulan *parent* berdasarkan *belief* pada *a*. Oleh karena itu ada dua hal dalam matrik *link*, yaitu: bagaimana menduga kebergantungan sebuah node pada himpunan *parent*-nya, dan bagaimana menuliskan dugaan tersebut dalam suatu matrik *link*.

Sebagai ilustrasi akan diperlihatkan matrik *link* untuk 3 *parent*. Matrik *link* untuk node Q berbentuk L[i,j], dimana $i \in \{0, 1\}$, $0 \le j < 2^3$. Artinya matrik *link* untuk node Q bernilai *true* (i=0) pada baris pertama dan Q bernilai *false* (i=1) pada baris kedua, sedangkan jumlah



Gambar 1. Model dasar yang diperluas

kolom ditunjukkan oleh banyaknya *j*, sehingga jumlah kolom untuk 3 *parent* A, B dan C ada sebanyak $2^3 = 8$ kolom, masing-masing kolom berkorespondensi dengan sebuah kombinasi tertentu dari nilai *parent*. Namakan kolom tersebut dari 0 sampai (2^3-1) atau dari 0 sampai 7, dan gunakan representasi biner dari nomor kolom tersebut sebagai nilai indeks untuk *parent*: A, B dan C. Baris kedua pada kolom 0 representasi binernya adalah 000_2 berkorespondensi dengan kasus dimana *A = false*, *B = false*, dan *C=false*, kolom 1 representasi binernya adalah 001_2 berkorespondensi dengan *A=false*, *B=false* dan *C=true*, kolom 2 representasi binernya adalah 010_2 berkorespondensi dengan A=*false*, *B=true*, dan *C=false*, ….., sedangkan kolom 7 representasi binernya adalah 111_2 yang berkorespondensi dengan kasus di mana semua *parent* bernilai *true*. Sedangkan nilai kolom j pada baris pertama merupakan negasi dari nilai kolom j dari baris kedua. Oleh karenanya suatu node dengan n *parent* mempunyai matrik *link* berukuran 2×2^n.

Dibawah ini akan diuraikan 5 bentuk matrik *link* pada node Q dengan 3 *parent*, yaitu matrik *link* untuk operator: *or, and, sum* dan *wtd*, dan matrik *link* pada node Q dengan satu *parent* untuk operator *not*. Operator *and, or,* dan *not* merupakan operator *Boolean* bila dugaan terhadap kombinasi *parent* untuk proposisi *a* bernilai 0 atau 1, sedangkan operator *sum*

dan *wtd* merupakan operator probabilistik karena dugaan terhadap *parent* dari proposisi *a* nilainya berada dalam interval (0,1).

Misalkan node *Q* mempunyai tiga *parent*, *A, B, C* dan misalkan istilah queri A berbobot *a* ditulis P(*A=true*)=*a*, juga P(*B=true*)=*b*, P(*C=true*)=*c*. Matrik *link* untuk operator-*or* dibahas sbb: Nilai node *Q* menjadi *true* bila sedikitnya satu dari *A, B* atau *C* bernilai *true* dan *false* bila *A*, *B* dan *C* semuanya *false*. Maka matrik *link* yang dikehendaki adalah:

L_{or} =

Baris pertama berkorespondensi dengan kasus dimana *Q=false* dan baris kedua berkorespondensi dengan *Q=true*. Kolom yang merepresentasikan sedikitnya salah satu dari A,B dan C *true* pada baris kedua (Q *true*)

Tabel 1. Tabel kebenaran untuk operator *or*

A	B	C	∨
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

terdapat di semua kolom kecuali kolom 0. Kolom 1 representasi binernya adalah (001_2). (001_2) berkorespondensi dengan nilai dari *parent* Q (istilah queri) untuk A bernilai *false*, B bernilai *false*, dan A bernilai *true*. Dengan cara yang sama berlaku pula untuk kolom 2 (010_2), kolom 3 (011_2), kolom 4 (100_2), kolom 5 (101_2), kolom 6 (110_2) dan kolom 7 (111_2). Hal ini menunjukkan bahwa *belief or* pada Q=*true* mempunyai informasi *diagnostic* seperti yang tercantum pada baris kedua dengan memperhitungkan nilai *predictive* yang diturunkan *parent*, misalnya kolom 3 baris 2 bernilai 1 (sebagai informasi *diagnostic*) karena *parent* menurunkan nilai "a(1-b)c" (sebagai komponen *predictive*), …dst.

Menentukan node Q bernilai *true* menurut aturan matrik *link* pada operator *or* tidak bertentangan dengan teori logika proposisi untuk *disjunction* dari A, B dan C. Dapat diperhatikan pada *truth table* untuk *disjunction (or,*atau dengan notasi ∨*)* dari A, B dan C (Tabel 1).

Menghitung *belief* untuk *Q=true,* adalah dengan menghitung peluang untuk setiap kombinasi yang mungkin dari variabel *parent* dan mengalikan peluang tersebut dengan elemen matrik pada baris kedua. Matrik *link* untuk operator *or* pada baris kedua yang mempunyai kombinasi *parent* bernilai 0 (*false*) hanya kolom 0 (000_2), sedangkan kolom lainnya mempunyai kombinasi *parent* bernilai 1 (*true*), yaitu kolom 1 sampai kolom 7 (atau baris 2 sampai baris 8 pada Tabel 1). Bila *P*(*A=true*)=*a*, maka *P*(*A=false*)=1-*a*. Hal ini berlaku juga untuk peluang B dan C, sehingga untuk menghitung besaran peluang *P*(*Q = true*):

$$P(Q=true) = 0(1-a)(1-b)(1-c) + 1(1-a)(1-b)c + 1(1-a)b(1-c) + 1(1-a)bc + 1a(1-b)(1-c) + 1a(1-b)c + 1ab(1-c) + 1abc$$
$$= 1-(1-a)(1-b)(1-c)$$
$$P(Q=false)=(1-a)(1-b)(1-c)$$

Matrik *link* untuk operator *and* diuraikan sbb: Nilai node Q adalah *true* jika semua A, B, dan C *true*. Pernyataan ini juga sesuai dengan aturan *conjunction* pada proposisi kalkulus. Node A, B dan C semua *true* jika binernya (111_2) yang merupakan bilangan 7 untuk angka berbasis sepuluh, jadi hanya kolom 7 yang bernilai *true*. Sedangkan kolom lainnya bernilai *false* karena sedikitnya satu dari kombinasi nilai antara A, B dan C bernilai *false*. Oleh karenanya bentuk matrik *link* untuk operator *and* sbb:

L_{and} =

Sehingga peluang *P*(*Q*=*true*):
P(*Q*=*true*) = *abc*
P(*Q*=*false*)=(1-*a*)(1-*b*)(1-*c)*
+(1-*a*)(1-*b)c*+(1-*a*)*b(1-c)*
+(1-*a*)*bc*+*a*(1-*b*)(1-c)
+a(1-*bc*+ *ab(1-c)*
=1-abc

Matrik *link* untuk operator-*not* uraiannya sbb: Operator-*not* pada node Q didefinisikan hanya untuk Q dengan *parent* tunggal. Misalkan *parent* dari Q adalah A, maka Q = *true* jika A = *false*, dan Q= *false* jika A=*true*, sehingga matrik untuk operator *not* sbb:

L_{not} [i,j], i$\in${0.1}, 0 ≤ j < 2

L_{not} =

P(*Q*=*true*) = 1-*a*
P(*Q*=*false*) = *a*

Matrik *link* untuk operator *sum* uraiannya sbb: Bentuk matrik *link sum* menspesifikasikan kepercayaan pada Q hanya tergantung pada sejumlah *parent* yang bernilai *true*. Jika j berkorespondensi dengan sejumlah kolom matrik *link* dimana m *parent true*, maka:

L_{sum}[1,j] = ,

(terdapat m dari n *parent* bernilai *true*)
Artinya baris kedua (Q *true*) untuk 3 *parent* mempunyai nilai matrik *link* pada kolom j=0 (000_2) sebesar 0 karena tidak ada *parent* bernilai *true*, nilai matrik *link* pada kolom j=1 (001_2) sebesar 1/3 karena mempunyai satu *parent* C bernilai *true*, nilai matrik *link* pada kolom j=2 (010_2) sebesar 1/3 karena 1 *parent* B yang bernilai *true*, nilai matrik *link* pada kolom j=3 (101_2) sebesar 2/3 karena mempunyai 2 *parent* A dan C bernilai *true*, ….dst.

Sedangkan untuk nilai kolom Q *false* pada baris pertama merupakan negasi dari nilai kolom tersebut pada baris kedua.

$L_{\boldsymbol{sum}}$ [0.j] =

Sehingga matrik *link-sum* dengan tiga *parent* dapat direpresentasikan sbb:

$L_{\boldsymbol{sum}}$ =

Evaluasi matrik *link sum* adalah:

P(Q=*true*)= (1-a)(1-b)c+ (1-a)b(1-c)

\+ (1-a)bc + a(1-b)(1-c)

\+ a(1-b)(1-c) + ab(1-c)+abc

=

P(Q=*false*)=(1-a)(1-b)(1-c) + (1-a)(1-b)c)

+ (1-a)b(1c)+ (1-a)bc

+ a(1-b)(1-c) + a(1-b)c+ ab(1-c).

= 1-

Sedangkan matrik *link* untuk operator *wtd* diuraikan sbb: Matrik *link* untuk operator *wtd* merupakan matrik *sum* yang berbobot. Jika semua bobot pada matrik *wtd* bernilai 1 maka matrik *wtd* sama dengan matrik *sum*. Untuk queri dengan pendekatan probabilistik masing-masing node *parent* dan node *children* mempunyai bobot dalam interval [0.1]. Dalam matrik *sum* kepercayaan terhadap Q tergantung pada node *parent* yang bernilai *true*. Node *parent* yang bernilai *true* diberi bobot dimana bobot yang lebih besar akan mempunyai lebih banyak pengaruh dalam *belief*. Jika w_a, w_b, w_c merupakan bobot dari node *parent* A, B dan C dan w_q bobot dari node queri Q, dimana $0 \le w_q \le 1$, sedangkan $t=w_a+w_b+w_c$ sehingga bentuk matrik *link* untuk *sum* yang berbobot dinotasikan sebagai operator "*wtd* " dapat dilihat pada Gambar 2.

Baris kedua dan kolom 0 (000_2) pada matrik L_{wtd} tidak ada *parent* yang bernilai *true*, pada kolom 1 (001_2) hanya C yang *true* sehingga C diberi bobot sebesar W_c dikalikan dengan bobot Q (w_q) dan dibagi dengan jumlah bobot dari ketiga node *parent* ($w_a+w_b+w_c=t$), kolom 3 (011_2) B dan C *true* dimana masing-masing diberi bobot sebesar w_b dan w_c, dikalikan dengan w_q dan dibagi dengan t, ……dst.

Gambar 2. Matrik L_{wtd}

Tabel 2. Matrik Kanonik

$Bel_{or}(Q)=1-(1-p_1)\times \ldots \times(1-p_n)$

$Bel_{and}(Q)=p_1\, p_2 \times \ldots\ p_n$

$Bel_{not}(Q)=1-p_1$

$Bel_{sum}(Q) =$

$Bel_{wtd}(Q) =$

+ a(1-b)(1-c) + a(1-b)c

+ ab(1-c)+ abc.

=

P(Q=*false*)= (1-a)(1-b)(1-c)

+ (1-a)(1-b)c

+ (1-a)b(1-c)

+ (1-a)bc + a(1-b)(1-c)

+ a(1-b)c + ab(1-c)

= 1 -

Evaluasi bentuk matrik *link wtd* sbb:

P(*Q*=*true*)= (1-a)(1-b)c

+ (1-a)b(1-c)

+ (1-a)bc

Untuk mengetahui peluang Q bernilai *true* matrik *link* dievaluasi dengan cara menghitung peluang untuk setiap kombinasi yang mungkin dari variabel *parent* dan mengalikan peluang tersebut dengan elemen matrik pada kolom yang sesuai dengan kombinasi *parent* pada baris kedua. Hasil evaluasi tersebut menghasilkan rumus untuk matrik kanonik. Dengan cara yang sama dapat dibuatkan matrik kanonik untuk *n* istilah queri, dan untuk menghemat "*time*" dan "*space"* queri dengan *n* istilah tidak dievaluasi dengan matrik *link* tetapi cukup dengan matrik kanonik seperti yang terlihat pada Tabel 2.

Catatan: Dari matrik kanonik yang digunakan untuk mengevaluasi queri maka queri berbentuk: "*not* A" dapat diimplimentasikan pada sistem temu-kembali dengan jaringan inferensi sementara pada metode lainnya ditabukan karena pada metode lain akan menampilkan hampir seluruh isi dokumen dalam koleksi [16].

Berdasarkan teori yang diuraikan sebelumnya maka algoritma berikut dapat digunakan dalam mengkoding program, terdiri dari 2 tahap yaitu proses evaluasi queri dan proses queri ganda.

Algoritma 1: evaluasi queri

1. Ambil ekspresi Jaringan Inferensi
2. Untuk setiap parent dari yang dievaluasi, ambil id-dok dan bobot
   //parent dari queri adalah istilah dokumen
   //parent dari information need adalah queri
3. Periksa operator dari yang dievaluasi
4. Jika parent lebih dari 2, lakukan penggabungan

dokumen

dari 2 parent yang dihubungkan oleh operator

5. Instanstiasi setiap dokumen dengan matrik kanonik

//instanstiasi dokumen default

//rumus (*or,sum,wtd*) disempurnakan pada

parent terakhir

//ambil bobot default untuk dokumen yang

tidak mempunyai pasangan

6. jika queri pertama, sorting dokumen terambil, print

Proses mengajukan queri berikutnya berlanjut jika dokumen terambil sebagai keluaran dari algoritma 1 belum memuaskan pemakai maka pemakai dapat menyempurnakan queri dengan menambahkan istilah queri yang diidentifikasi pemakai setelah melihat keluaran dokumen hasil algoritma 1 atau mengkombinasikan menjadi queri ganda.

Algoritma disusun untuk dua pilihan dalam menyempurnakan queri:

- Pilihan pertama (q) adalah evaluasi queri menurut algoritma 1, mengikuti aturan pada teori Jaringan Inferensi dimana semua queri menjadi *parent* bagi node I. Jadi masukan pada modul ini adalah semua operand (data queri) sebelumnya dan sebuah formulasi queri berikutnya, sedangkan keluarannya adalah himpunan dokumen yang terambil.
- Pilihan kedua (I), untuk queri ganda, menurut jaringan dengan topologi seperti pada Gambar 1, yang merupakan pengembangan dari model dasar. Pilihan I untuk mengkombinasikan queri antara tipe *Boolean* yang mengandung lebih dari satu operator dan tipe pendekatan probabilistik (alami). Sebagai masukan adalah data terakhir dari hasil evaluasi permintaan informasi (I) yang diletakkan dalam operand pertama, dan satu queri berikutnya yang hasil evaluasinya diletakkan dalam operand kedua. Keluarannya adalah himpunan dokumen yang terambil. Diharapkan urutan atas dari dokumen terambil adalah dokumen yang relevan dengan queri pemakai.

Modul evaluasi bukan bagian dari sistem temu-kembali informasi, tetapi dirancang untuk melihat sejauh mana efektifitas sistem yang dikembangkan. Efektifitas sistem temu-kembali dilihat dari tabel *Recall-Precision*. *Recall* adalah proporsi dari dokumen relevan yang terambil, sedangkan *Precision* adalah proporsi dokumen terambil yang relevan [6, 10]. Misalkan a+b+c+d

Algoritma 2: queri ganda

Selagi dokumen terambil belum memuaskan pemakai

// pilih "q" atau "I"

Ambil ekspresi queri berikutnya

Kirim ke modul proses queri

Evaluasi dengan algoritma 1

Letakkan hasilnya dalam operand berikutnya

Tentukan tipe operator untuk node I

Jika pilihan q

Ambil data semua operand sebagai parent

Evaluasi semua operand dengan algoritma 1

Else

Letakkan data I sebelumnya dalam operand1

Letakkan data queri terakhir dalam operand2

Evaluasi kedua operand dengan algoritma 1

Sorting dokumen-dokumen yang terambil

Print dokumen yang terambil

Tabel 3. Nilai presisi untuk alpha

	Precision				
Rc	α=0.0	α=0.1	α=0.2	α=0.3	α=0.4
0.1	5.155	5.078	5.089	4.601	5.571
0.2	4.917	4.872	4.883	4.337	5.367
0.3	4.108	4.578	4.732	4.501	4.917
0.4	3.710	4.501	4.721	4.357	5.075
0.5	3.517	3.588	3.742	3.725	4.311
0.6	1.027	3.496	3.637	3.620	4.227
0.7	1.046	3.093	3.157	3.341	3.665
0.8	1.061	3.174	3.239	3.408	3.655
0.9	0.917	2.670	2.727	2.905	3.034
1.0	0.917	2.501	2.516	2.583	2.622
Rata	2.638	3.755	3.844	3.738	**4.244**

merupakan jumlah dokumen dalam koleksi dokumen, jika kondisi terhadap queri yang diajukan adalah a+b bagian dokumen yang terambil, b+c bagian dokumen yang relevan dan b adalah bagian dokumen relevan yang terambil maka:

Rc = , Pc =

Dalam penelitian ini nilai *Precision* (Pc) dihitung disetiap 10 titik *Recall* (Rc). Level *Recall* ditentukan disetiap kenaikan 10 % dari jumlah dokumen relevan yang terambil, dan nilai Pc dihasilkan dengan menghitung rata-rata nilai Pc disetiap titik *Recall*.

Metode temu-kembali yang dikembangkan diaplikasikan pada sekumpulan queri, setiap queri harus diketahui relevansi terhadap dokumen dalam koleksi dokumen. Penilaian relevansi dilakukan secara manual oleh penulis dengan menyesuaikan queri dengan dokumen yang ada dalam koleksi dokumen. Sebaiknya penilaian diberikan oleh pakar yang ahli dibidangnya atau oleh *real user*.

Proses evaluasi dirancang dengan 2 subproses, yakni (1) Poses menghitung *Precision*. Masukan dari modul ini ada 2 berkas, yaitu berkas data relevansi dokumen dengan queri yang diuji cobakan, dan berkas dokumen yang terambil dari queri tersebut. Berkas dokumen yang terambil ini bernilai 1 jika dokumen yang bersangkutan dianggap relevan dan bernilai 0 jika sebaliknya. Keluarannya adalah nilai *Precision* (Pc) disetiap 10 titik *Recall* (Rc). Proses ini dilakukan untuk setiap queri sehingga semua queri dalam kumpulan queri habis diselidiki.

Algoritma menghitung Pc:

```
Hitung banyaknya dokumen relevan  pada setiap queri
Untuk setiap kumulatif 10 persen sebagai nilai
        Rc dari banyaknya dokumen relevan, lakukan:
Hitung presentase dokumen relevan yang terambil
        Sebagai nilai Pc untuk titik Rc ybs
Ulangi untuk setiap queri dan setiap tehnik pencarian
        yang berbeda dalam Jaringan Inferensi
```

(2) Proses menghitung jumlah Pc di setiap titik Rc untuk semua queri. Masukannya adalah beberapa nilai Pc disetiap titik Rc yang dihasilkan dari subproses pertama, dan keluarannya adalah rata-rata nilai Pc di setiap titik Rc.

Algoritma menghitung rata-rata Pc:

```
Untuk setiap formulasi yang akan dibandingkan
 Hitung jumlah nilai Pc disetiap titik Rc
```

pada sejumlah queri yang akan dibandingkan.

Tabel 4. Nilai presisi *default*

	Precision					
Rc	0.0	0.1	0.2	0.3	0.4	0.5
0.1	5.23	5.19	5.19	5.48	5.57	6.04
0.2	4.99	4.95	4.95	5.25	5.36	5.79
0.3	3.91	4.23	4.23	4.37	4.92	4.79
0.4	2.72	4.27	4.14	4.48	5.07	4.80
0.5	2.36	3.29	3.20	3.63	4.31	3.82
0.6	1.66	3.13	3.03	3.48	4.23	3.66
0.7	0.89	2.97	2.96	3.25	3.66	3.26
0.8	0.91	3.06	3.04	3.30	3.65	3.00
0.9	0.92	2.76	2.76	2.93	3.03	2.72
1.0	0.92	2.51	2.51	2.55	2.62	2.42
Rt	2.45	3.54	3.60	3.87	4.24	4.03

3. Hasil dan Pembahasan

Akan diperiksa efisiensi sistem temu-kembali berbasis jaringan inferensi pada kumpulan dokumen terhadap sekumpulan queri yang telah diketahui relevansi dokumennya dalam kumpulan dokumen di Fasilkom UI. Pertama-tama akan diperiksa nilai alpha dan default pada sistem, kemudian queri tunggal (*Boolean* atau dengan pendekatan probabilistik) kemudian queri ganda.

Diperiksa nilai α yang sesuai untuk fungsi pembobotan $P(r_i|d_i=true) = \alpha+(1-\alpha)\times ntf_{ij}\times nidf_i$. $\alpha\in[0.0, 0.5)$ pada koleksi dokumen. Pilih besaran α pada titik {0.0, 0.1, 0.2, 0.3, 0.4}.

Implementasi kumpulan queri untuk setiap besaran α dengan sistem yang dikembangkan menghasilkan tabel *precision* dan *recall* seperti yang terlihat pada Tabel 3. Besaran α=0.4 adalah besaran yang paling sesuai untuk diaplikasikan pada fungsi pembobotan istilah dokumen karena menghasilkan rata-rata presisi yang paling besar yaitu sebesar 4.244 sehingga untuk berikutnya berkas pembalikan dokumen yang dihasilkan dari implementasi α=0.4 yang digunakan.

Setelah nilai alpha ditentukan pada $P(r_i|d_j=true)=0.4+(1-0.6)\times ntf_{ij}\times nidf_i$, selanjutnya diperiksa $P(r_i|d_j=false)=\beta$, $\beta\in[0.0,0.5)$.

Tabel 4 menunjukkan adanya peningkatan presisi dari β = 0.0 sampai β=0.4 dan penurunan presisi dari β=0.4 ke β=0.5. artinya $P(r_i \mid d_j= false)$=0.4 adalah yang terbaik, ini sesuai dengan saran (Davis, 1995) dimana bila $P(r_i \mid d_j=false)$=0.4 maka $P(r_i|d_j=true)$ hendaknya berada dalam (0.5,1].

Tabel 5 dihasilkan dari queri tunggal untuk semua kumpulan queri. Kolom *or* pada Tabel 5 dihasilkan

Tabel 5. Queri tunggal

	Presisi			
Rc	*or*	*and*	Bool	*sum*
0.1	0.628	1.308	1.333	5.404
0.2	0.795	1.142	1.213	5.154
0.3	0.943	1.163	1.016	4.685
0.4	0.698	1.047	1.141	4.636
0.5	0.539	0.905	1.268	3.567
0.6	0.538	0.867	1.321	3.391
0.7	0.557	0.748	1.216	3.036
0.8	0.568	0.741	1.136	3.100

0.9	0.576	0.662	0.970	2.649
1.0	0.586	0.661	0.780	2.425
Rata	0.543	0.924	1.139	3.805

dari queri tunggal dengan operator *or*, begitu juga untuk kolom *and*, dan kolom *sum* masing-masing dihasilkan dari queri tunggal dengan operator *and* dan *sum*. Sedangkan kolom *Bool* dihasilkan dari queri tunggal dengan operator *and* dan *or*. Dari Tabel 5 terlihat bahwa queri tunggal dengan pendekatan proba-bilistik yang diwakili oleh operator *sum* menghasilkan presisi yang paling tinggi, artinya dokumen terambil yang relevan berada diurutan sebelah atas.

Queri ganda dihasilkan dengan mengajukan semua queri secara bertahap menurut Gambar 1. Pada Tabel 6 kolom *Gbol* dihasilkan dari queri ganda dengan operator *and* dan *or*, sedangkan kolom G*and*, G*or*, G*sm* dan G*wtd*$_{28}$ dihasilkan dari queri ganda masing-masing dengan operator *and*, *or*, *sum* dan *wtd*, tanda subskrip setelah penulisan *wtd* menunjukkan bobot yang diberikan untuk queri *Boolean* sebesar 0.2 dan untuk queri probabilistik sebesar 0.8.

Dari Tabel 5 dan Tabel 6 terlihat bahwa queri ganda menghasilkan rata-rata presisi yang lebih besar dari pada queri *Boolean* atau probabilistik, artinya dokumen terambil dengan queri ganda akan mengurut dokumen yang relevan dengan queri dibagian sebelah atas sehingga memberi kemudahan kepada pemakai untuk menspesifikasi dokumen yang dikehendakinya.

4. Kesimpulan

Dari hasil simulasi komputer pada studi ini dapat disimpulkan bahwa sistem yang dikembangkan dapat memeriksa besaran yang tepat untuk α dan β pada fungsi pembobotan istilah untuk koleksi dokumen yang digunakan. Menyediakan operator yang lebih banyak, yaitu: operator *and, or, not, sum* dan *wtd* yang pada sistem temu-kembali lainnya hanya tersedia operator *and, or* dan *not*. Berfungsi sebagai penghubung antara pemakai dengan kumpulan dokumen karena formulasi queri dapat diajukan secara bertahap sehingga pemakai dapat mengidentifikasi istilah penting dalam dokumen relevan yang akan diajukan pada formulasi queri berikutnya. Sistem ini memberikan keleluasaan pada

Tabel 6. Queri ganda

	Presisi				
Rc	Gbol	Gand	Gor	Gsm	Gwtd$_{28}$
0.1	5.57	5.66	6.14	6.14	6.17
0.2	5.37	5.42	5.90	5.90	5.97
0.3	4.92	5.13	5.73	5.73	5.77
0.4	5.07	5.16	5.28	5.28	5.40
0.5	4.31	4.25	4.24	4.24	4.15
0.6	4.23	4.13	4.12	4.12	4.03
0.7	3.66	3.58	3.62	3.59	3.43
0.8	3.65	3.66	3.69	3.67	3.51
0.9	3.03	3.06	3.09	3.07	2.92
1.0	2.62	2.66	2.66	2.66	2.58
Rata	4.22	4.27	4.44	4.44	4.39

pemakai dalam memformulasikan queri, baik sebagai queri *Boolean* atau probabilistik maupun kombinasi antara keduanya (ganda). Sistem ini memberi kemudahan pada pemakai untuk mengidentifikasi dokumen terambil yang relevan seperti yang dinginkannya, karena dokumen relevan berada dibagian sebelah atas. Studi ini memperlihatkan bahwa queri ganda meningkatkan kinerja sistem jika dibandingkan dengan queri *boolean* atau queri dengan pendekatan probabilistik.

Ucapan Terima Kasih

Ucapan terima kasih disampaikan kepada Bapak Zaenal Arifin Hasibuan, Ph.D atas peran beliau memperkenalkan teori tentang *Information Retrieval Systems* sehingga makalah ini dapat terwujud.

Daftar Acuan

[1] A. Toffler, Future Shock, Pan Books Ltd, London, 1970.

[2] J. Pearl, Probabilistic Reasoning in Intelligent Sistems: Network of Plasible Inference, Morgan Kaufmann Publishers Inc., California, 1978.
[3] G. Salton, E. Fox, H. Wu, Communications of the ACM 26 (1983) 1022.
[4] N. Fuhr, Information Processing & Management 25 (1989) 55.
[5] G. Salton, A. Wong, C.S. Yang, Communications of the ACM 18 (1975) 613.
[6] V. Tahani, Information Processing and Management. 15 (1976) 177.
[7] E. Fox., S. Betrabet, M. Koushik, In: W.B. Frakes, R. Baeza-Yates (Eds), Information Retrieval: Data Structure & Algorithms, Prentice Hall, New York, 1976.
[8] H.R. Turtle, PhD Thesis, University of Massachusetts, USA, 1990.
[9] M. Iivonen, Information Processing & Management 31 (1995) 173.
[10] R. Magdalena, Skripsi Sarjana, Fakultas Ilmu Komputer, Universitas Indonesia, Indonesia, 1996.
[11] H.R. Turtle, W.B. Croft. ACM Transactions on information Sistems 9 (1991) 187.
[12] G. Salton, The Journal of Documentation 35 (1979) 1.
[13] D. Harman, In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pittsburgh, 1992.
[14] J. Belkin, C. Cool, W.B. Croft, J.P. Callan, In Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pittsburgh, 1993.
[15] G. Salton, Automatic Text Processing, Addison-Wesley Longman Publishing Co. Inc., Boston, 1989.
[16] Y. Wisnani, Tesis, Program Studi Ilmu Komputer, Fakultas Ilmu Komputer, Universitas Indonesia, Indonesia, 1998.