MULTIPLE SEQUENCE ALIGNMENT MENGGUNAKAN *HIDDEN MARKOV MODEL*

Afiahayati [1)] , Sri Mulyana [2)]
[1,2)] Program Studi Ilmu Komputer, Universitas Gadjah Mada Yogyakarta
email : af1a@mail.ugm.ac.id, smulyana@mail.ugm.ac.id

Abstrak

Mudah dan murahnya proses pengumpulan data biologi molekuler saat ini menyebabkan ukuran basis data genetika meningkat dengan pesat. Hal ini meningkatkan kebutuhan akan alat bantu komputasi untuk menganalisa data tersebut. Salah satu task dasar dalam menganalisa data biologi molekuler adalah Multiple Sequence Alignment. Program Multiple Sequence Alignment yang sering digunakan oleh praktisi biomolekuler adalah ClustalX yang menggunakan metode komputasi progressive pairwise alignment.

Salah satu metode yang saat ini banyak dikaji untuk menghasilkan Multiple Sequence Alignment adalah Hidden Markov Model. Hidden Markov Model cocok digunakan dalam Multiple Sequence Alignment karena Multiple Sequence Alignment dapat dipandang sebagai masalah pengenalan pola. Hidden Markov Model menggunakan algoritma pembelajaran Baum-Welch untuk mengestimasi parameter-parameter dalam HMM dan algoritma Viterbi untuk melakukan alignment dari unaligned sequence.

Pada penelitian ini dilakukan eksperimen untuk menerapkan Hidden Markov Model dalam menghasilkan Multiple Sequence Alignment dari sequence protein yang belum ter-align dan dilakukan pengujian menggunakan data sequence protein BaliBASE 3.0 dengan membandingkan hasil alignment yang menerapkan Hidden Markov Model dengan hasil alignment program ClustalX. Hasil eksperimen menunjukkan bahwa implementasi Hidden Markov Model pada Multiple Sequence Alignment memiliki performa lebih baik pada data sequence yang memiliki identity tinggi dan mengalami penurunan perfoma pada data sequence yang panjang dan data sequence yang memiliki banyak noise seperti N/C terminal extension atau insertion.

Keyword: *BaliBASE 3.0, Baum-Welch, Hidden Markov Model, Multiple Sequence Alignment, Viterbi.*

1. PENDAHULUAN

Kemudahan dan murahnya pengumpulan data genetika menyebabkan basisdata genetika semakin membesar. Peningkatan jumlah data yang pesat ini menyebabkan analisa data secara manual tidak lagi efisien. Dibutuhkan komputasi untuk membantu analisa data sehingga dapat mengekstrak suatu informasi penting dari sekumpulan data genetika yang berjumlah besar. Salah satu task dasar dalam menganalisa data biologi molekuler adalah *Multiple Sequence Alignment*. *Multiple sequence alignment* adalah dasar yang penting bagi *task* lain dalam bioinformatika seperti rekonstruksi pohon filogenetik dan *sub-family classification*.

Metode yang umum digunakan untuk menghasilkan *Multiple Sequence Alignment* adalah *dynamic programming*. Untuk n *sequence*, metode ini membangun matriks n-dimensi. Ruang pencarian meningkat secara eksponensial dengan bertambahnya n dan metode ini juga bergantung pada panjang *sequence*. *Multiple Sequence Alignment* memerlukan metode yang lebih baik daripada *dynamic programming* karena *dynamic programming* memerlukan biaya komputasi yang mahal. Metode *dynamic programming* sekarang hanya digunakan apabila dibutuhkan hasil *alignment* yang berkualitas sangat tinggi dengan jumlah *sequence* yang sedikit, dan sebagai standard pembanding untuk mengevaluasi teknik heuristik yang baru. Beberapa metode lain telah diimplementasikan dalam *Multiple Sequence Alignment*. Program *Multiple Sequence Alignment* yang sering digunakan oleh praktisi biomolekuler adalah ClustalX yang menggunakan metode heuristik *progressive pairwise alignment.* Salah satu metode yang cukup baik dan beberapa tahun belakangan banyak dikembangkan adalah metode *Hidden Markov Model*.

Awalnya, *Hidden Markov Model* dipakai dalam bidang *Speech Recognition* (Gupta, 2004). HMM mulai digunakan dalam bioinformatika sejak diperkenalkan oleh Krogh et al. (Gupta, 2004). HMM adalah model statistika, dimana sistem yang dimodelkan diasumsikan merupakan proses Markov dengan parameter yang tidak diketahui (Koski, 2001). Tantangan dalam metode ini adalah menentukan parameter-parameter tersembunyi yang tidak diketahui dari keluaran yang dihasilkan oleh model tersebut. Suatu proses disebut proses Markov apabila memiliki properti Markov yaitu distribusi probabilitas *state* berikutnya ditentukan oleh *state* saat ini dimana probabilitas *state* saat ini ditentukan oleh *state* sebelumnya. HMM banyak digunakan untuk masalah-masalah pengenalan pola (*pattern recognition*), seperti *speech recognition*. MSA dapat dipandang sebagai masalah pengenalan pola, yaitu pengenalan dan pembandingan pola *sequence* protein. Oleh karena itu, HMM dapat digunakan untuk memodelkan masalah *Multiple Sequence Alignment.*

Pada penelitian ini dilakukan eksperimen untuk menerapkan *Hidden Markov Model* dalam menghasilkan *Multiple Sequence Alignment* dari *sequence* protein yang belum ter-*align* dan dilakukan pengujian menggunakan data *sequence* protein BaliBASE 3.0 dengan membandingkan hasil *alignment* yang menerapkan

Hidden Markov Model dengan hasil *alignment* program ClustalX untuk mengetahui perfoma protipe perangkat lunak yang menerapkan *Hidden Markov Model* terhadap kumpulan *sequence* protein dengan kriteria – kriteria tertentu yang merepresentasikan permasalahan-permasalahan nyata yang sering dihadapi ketika melakukan *alignment* terhadap *sequence* protein. Kriteria tersebut diantaranya panjang *sequence*, tingkat *identity* (kesamaan) diantara *sequence*, keberadaan *internal insertion* dan keberadaan *N/C terminal extension* di dalam *sequence*.

Dengan dilakukan penelitian ini, diharapkan dapat menjadi dorongan untuk mengembangkan metode baru yang lebih baik dalam *Multiple Sequence Alignment* dan meningkatkan perkembangan studi bioinformatika di Indonesia.

2. TINJAUAN PUSTAKA

2.1. Hidden Markov Model

Sebuah HMM menggabungkan dua atau lebih rantai Markov dengan hanya satu rantai yang terdiri dari *state* yang dapat diobservasi dan rantai lainnya membentuk *state* yang tidak dapat diobservasi (*hidden*), yang mempengaruhi hasil dari *state* yang dapat diobservasi. Probabilitas dari satu *state* ke *state* lainnya dinamakan *transition probability*. Setiap *state* mungkin dibentuk oleh sejumlah elemen atau simbol. Untuk *sequence* nukleotida (*sequence* gen), terdapat empat buah kemungkinan simbol – A, T, G, dan C – dalam setiap *state*. Untuk *sequence* asam amino, terdapat dua puluh buah simbol. Nilai probabilitas yang berasosiasi dengan setiap simbol dalam setiap *state* disebut *emission probability*. Untuk menghitung probabilitas total dari suatu jalur dalam model, baik *transition probability* maupun *emission probability* yang menghubungkan semua *hidden state* dan *state* yang dapat diobservasi harus dimasukkan dalam perhitungan (Gupta, 2004).

Sebuah *Hidden Markov Model* dikarakteristikkan dengan parameter berikut (Rabiner, 1989):

1. N, jumlah *state* dalam model.
2. M, jumlah simbol pengamatan yang dimiliki setiap *state*.
3. $A = \{a_{ij}\}$, $a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i)$, himpunan distribusi kemungkinan perpindahan *state* (*transition probability*) .
4. $B = \{b_j(k)\}$, $b_j(k) = P(v_k$ pada $_t \mid q_t = S_j)$, himpunan distribusi kemungkinan simbol pengamatan pada *state j* (*emission probability*).
5. $\pi = \{\pi_i\}$, $\pi_i = P(q_1 = S_i)$, himpunan distribusi kemungkinan *state* awal.

Bentuk ringkas dari HMM adalah

$$\lambda = (A, B, \pi) \qquad (2.1)$$

2.1.1 Persoalan Utama dalam HMM

Terdapat tiga persoalan utama yang harus dipecahkan agar HMM dapat digunakan dalam suatu aplikasi nyata. Persoalan tersebut adalah (Rabiner, 1989):

1. Diberikan model $\lambda = (A, B, \pi)$, bagaimana menghitung $P(O \mid \lambda)$, yaitu kemungkinan ditemuinya rangkaian pengamatan $O = O_1, O_2, ..., O_T$.
2. Diberikan model $\lambda = (A, B, \pi)$, bagaimana memilih rangkaian *state* $I = i_1, i_2,...,i_T$ sehingga $P(O, I \mid \lambda)$, kemungkinan gabungan rangkaian pengamatan $O = O_1, O_2, ..., O_T$ dan rangkaian *state* jika diberikan model, maksimal.
3. Bagaimana mengubah parameter HMM, $\lambda = (A, B, \pi)$ sehingga $P(O \mid \lambda)$ maksimal.

Persoalan 1 dan 2 dapat dilihat sebagai persoalan analisis sedangkan persoalan 3 sebagai persoalan sintesis (atau disebut identifikasi model atau pelatihan).

2.1.2 Solusi Persoalan 1

Persoalan 1 dapat diselesaikan dengan menggunakan algoritma yang dinamakan prosedur maju-mundur (*forward-backward procedure*) (Rabiner, 1989). Pertama, dijelaskan prosedur *forward*, diasumsikan variabel maju $\alpha_t(i)$ didefinisikan sebagai:

$$\alpha_t(i) = P(O_1,O_2,...,O_t,\ q_t = S_i \mid \lambda) \qquad (2.2)$$

yaitu kemungkinan rangkaian pengamatan parsial hingga waktu t dan berada pada *state* S_i pada waktu t, jika diberikan model λ. Maka $\alpha_t(i)$ dapat dihitung dengan induksi sebagai berikut:

1. Inisialisasi

$$\alpha_t(i) = \pi_i b_i(O_1),\ 1 \le i \le N \qquad (2.3)$$

2. Induksi

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(\text{[illegible]} -1 \quad 1 \leq j \leq N \qquad (2.4)$$

3. Terminasi

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad (2.5)$$

Langkah pertama merupakan inisialisasi, langkah induksi merupakan langkah yang paling utama pada prosedur forward. P(O|λ) dapat dicari dengan menjumlahkan variabel maju dengan t=T dari semua state. P(O|λ) merupakan probabilitas model menghasilkan rangkaian pengamatan $O = O_1, O_2, ..., O_T$.

Dengan cara yang sejenis dapat didefinisikan variabel mundur $\beta t(i)$ sebagai berikut:

$$\beta t(i) = P(O_{t+1}, O_{t+2}, ..., O_T \mid q_t = S_i, \lambda) \quad (2.6)$$

yaitu kemungkinan rangkaian pengamatan dari t+1 hingga T jika diberikan *state* S_i pada waktu t dan model λ. $\beta_t(i)$ dapat diselesaikan sebagaimana $\alpha_t(i)$.

1. Inisialisasi

$$\beta_T(i) = 1, \ 1 \le i \le N \quad (2.7)$$

2. Induksi

$$\beta_t(i)=\sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad t=T-1,T-2,...,1 \qquad 1 \leqslant i \leqslant N \quad (2.8)$$

Variabel mundur akan digunakan pada persoalan 3, melakukan estimasi nilai parameter – parameter HMM.

2.1.3 Solusi Persoalan 2

Pada persoalan ini akan dicari rangkaian *state* $I = i_1, i_2, ..., i_T$ sedemikian hingga kemungkinan kemunculan rangkaian pengamatan $O=O_1,O_2,...,O_T$ menjadi maksimal. Atau dengan kata lain mencari I yang memaksimalkan $P(O, I \mid \lambda)$. Salah satu solusi penyelesaian persoalan ini adalah dengan menggunakan algoritma Viterbi (Rabiner, 1989). Pertama-tama didefinisikan:

$$\delta_{t+1}(j)= \max_{q_1 q_2 \cdots q_{t-1}} P[q_1 q_2 \ldots q_t = i, O_1 O_2 \ldots O_t | \lambda] \quad (2.9)$$

sebagai probabilitas tertinggi pada waktu t dan berakhir di *state* S_i.
Algoritma Viterbi sebagai berikut :

1. Inisialisasi

$$\delta_1(i)=\pi_i b_i(O_1) \qquad \psi_1(i)=0 \qquad 1 \leqslant i \leqslant N \quad (2.10)$$

2. Rekursi

$$\delta_t(j)= \max_{1 \leqslant i \leqslant N} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \qquad 2 \leqslant t \leqslant T \qquad 1 \leqslant j \leqslant N \quad (2.11)$$

$$\Psi_t(j)= \operatorname*{argmax}_{1 \leqslant i \leqslant N} [\delta_{t-1}(i) a_{ij}] \qquad 2 \leqslant t \leqslant T \qquad 1 \leqslant j \leqslant N \quad (2.12)$$

3. Terminasi

$$P^* = \max_{1 \leqslant i \leqslant N} [\delta_T(i)] \quad (2.13)$$

4. Penelusuran balik rangkaian *state* yang optimal

$$\vec{q_t} = \psi_{t+1}(\vec{q_{t+1}}) \qquad t=T-1,T-2,...,1 \quad (2.14)$$

2.1.4 Solusi Persoalan 3

Persoalan 3 merupakan persoalan yang paling sulit, bisa dikatakan sebagai persoalan pelatihan yang digunakan untuk menghasilkan parameter model A, B, dan π optimal sehingga dapat dengan baik merepresentasikan rangkaian observasi yang terjadi. Kriteria optimal adalah memaksimalkan probabilitas rangkaian pengamatan, $P(O \mid \lambda)$, dengan diberikan model, $\lambda(A, B, \pi)$. Tidak ada pendekatan analitik untuk permasalahan ini, namun terdapat prosedur iteratif seperti metode Baum-Welch yang dapat digunakan (Rabiner, 1989). Untuk mendeskripsikan formula pelatihan secara matematis, diasumsikan $\xi_t(i, j)$ sebagai probabilitas berada pada *state i* pada waktu t, dan *state j* pada waktu t+1, diberikan model dan rangkaian pengamatan:

$$\xi t\,(i,j) = P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda) \quad (2.16)$$

Dengan menggunakan definisi variabel maju dan mundur, persamaan di atas dapat ditulis dalam bentuk:

$$\xi_t(i,j)=\frac{(\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j))}{(P(O|\lambda))} \quad (2.17)$$

$$\xi_t(i,j)=\frac{(\alpha_t(i)\,a_{ij}\,b_j(O_{t+1})\,\beta_{t+1}(j))}{(\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)\,a_{ij}\,b_j(O_{t+1})\,\beta_{t+1}(j))} \tag{2.18}$$

P(O|λ) merupakan nilai probabilitas model λ memberikan sequence O. Biasanya P(O|λ) sering diberi nilai 1, nilai harapan model λ memberikan sequence O. Menjumlahkan $\xi t\,(i, j)$ pada $1 \le t \le T-1$ menghasilkan jumlah perpindahan dari *i* ke *j* yang diharapkan. Untuk kebutuhan pelatihan, didefinisikan probabilitas berada di *state i* pada waktu *t*, diberikan rangkaian pengamatan dan model sebagai $\gamma_t(i)$

$$\gamma_t(i)=\frac{(\alpha_t(i)\,\beta_t(i))}{(P(O|\lambda))}=\frac{(\alpha_t(i)\beta_t(i))}{(\sum_{i=1}^{N}\alpha_t(i)\,\beta_t(i))} \tag{2.19}$$

Selanjutnya, $\gamma_t(i)$ dan $\xi t\,(i, j)$ dapat dihubungkan dengan menjumlahkan $\xi t\,(i, j)$ untuk semua *j*:

$$\gamma_t(i)=\sum_{j=1}^{N}\xi_t(i,j) \tag{2.20}$$

Menjumlahkan , $\gamma_t(i)$ sepanjang waktu memberikan jumlah *state i* dikunjungi. Jika waktu *T* tidak dimasukkan, dengan kata lain menjumlahkan sepanjang $1 \le t \le T-1$, ini memberikan jumlah perpindahan dari *state i*.

$$\sum_{t=1}^{T-1}\gamma_t(i) \quad = \text{jumlah perpindahan yang diharapkan dari } S_i \tag{2.21}$$

$$\sum_{j=1}^{N}\xi_t(i,j) \quad = \text{jumlah perpindahan yang diharapkan dari } S_i \text{ ke } S_j \tag{2.22}$$

Menggunakan formula di atas, dapat didefinisikan formula untuk melakukan estimasi terhadap nilai – nilai parameter HMM

$$\overline{a_{ij}}=\frac{(\textit{jumlah perpindahan yang diharapkan dari Si ke Sj})}{(\textit{jumlah perpindahan yang diharapkan dari Si})}$$

$$\overline{a_{ij}}=\frac{(\sum_{t=1}^{N}\xi_t(i,j))}{(\sum_{t=1}^{T-1}\gamma_t(i))}$$

$$\overline{b_j}(k)=\frac{(\textit{jumlah frekuensi yang diharapkan pada state j dan menghasilkan simbol Vk})}{(\textit{jumlah frekuensi pada state j})} \tag{2.23}$$

$$\overline{b_j}(k)=\frac{(\sum_{t=1}^{T-1}\gamma_t(i))}{(\sum_{t=1}^{T-1}\gamma_t(i))} \tag{2.24}$$

2.2. *Multiple Sequence Alignment*

Multiple Sequence Alignment adalah *sequence alignment* dari tiga atau lebih *sequence* biologis, umumnya protein, DNA, atau RNA. Misalnya *multiple sequence alignment* protein, maka elemen *sequence* berupa asam amino. MSA membuat elemen dengan huruf yang sama pada *sequence* yang berbeda sedapat mungkin berada pada kolom yang sama. Pada umumnya, kumpulan *sequence* yang menjadi masukan diasumsikan memiliki hubungan evolusioner, yaitu hubungan silsilah keturunan atau hubungan kesamaan leluhur. Dengan MSA, dapat diketahui kesamaan homologi antar *sequence* dan dapat dilakukan analisis filogenetik untuk mengetahui asal hubungan evolusioner yang ada. Dari MSA dapat diketahui adanya mutasi yang dapat dilihat sebagai adanya perbedaan huruf pada satu kolom, dan adanya *insertion* atau *deletion* yang terlihat sebagai *gap* pada satu atau lebih *sequence* pada *alignment* (Colton, 2007).

MSA juga dikenal sebagai proses melakukan *alignment* pada sekumpulan *sequence*. Karena proses *alignment* pada tiga atau lebih *sequence* dengan panjang yang relevan secara biologis hampir mustahil dilakukan secara manual, algoritma komputasi digunakan untuk menghasilkan dan menganalisis *alignment*. Kebanyakan program MSA menggunakan metode heuristik daripada *global optimization* karena untuk mencapai *global optimization* memerlukan biaya komputasi yang mahal.

2.3. *Multiple Sequence Alignment* Menggunakan *Hidden Markov Model*

Hidden Markov Model yang digunakan memiliki struktur berulang yang terdiri atas tiga tipe *state* yaitu *match*, *insert*, dan *delete* (Krogh et al., 1992). *State-state* ini memiliki probabilitas yang berbeda-beda pada tiap posisi. Tiap *match state* berkorespondensi dengan sebuah kolom pada *multiple alignment* dan dinotasikan dengan kotak segi empat pada diagram. *Alignment* disebut trivial jika pada model hanya terdapat *match state*, karena tidak ada pilihan transisi lain selain menuju *match state* berikutnya. Tiap *match state* memiliki *emission probability*, $P(x_j|m_i)$, yaitu kemungkinan *match state* m_i mengeluarkan simbol x_j.

2.3.1. Algoritma *Baum Welch*

Parameter-parameter pada HMM seperti *transition probability* dan distribusi asam amino (*emission probability*) dapat dipilih secara manual berdasarkan *alignment sequence* protein yang ada, atau dari informasi struktur tiga dimensi protein. Parameter – parameter HMM diestimasi untuk menemukan model yang paling baik dalam mendeskripsikan suatu kumpulan *sequence*(data *training*).

Pada penelitian ini, parameter-parameter pada HMM akan ditentukan secara otomatis melalui proses pembelajaran data *sequence* protein yang belum ter-*align* menggunakan algoritma Baum-Welch. Inti dari pendekatan ini adalah menemukan model yang paling baik untuk mendeskripsikan suatu kumpulan *sequence* (*training set*).

Langkah – langkah dalam prosedur *forward – backward* untuk mengestimasi parameter-parameter dari suatu HMM adalah sebagai berikut (Krogh et al, 1992):

1. Menentukan model awal, menentukan panjang model, menentukan struktur model, memberikan suatu nilai awal (acak) untuk setiap *transition probability* dan *emission probability* dari HMM.
2. Menghitung probabilitas maju (*forward probability*) setiap *state* dari semua kemungkinan jalur perpindahan antar-*state* dengan menggunakan prosedur *forward*.
3. Menghitung probabilitas mundur (*backward probability*) setiap *state* dari semua kemungkinan jalur perpindahan antar-*state* dengan menggunakan prosedur *backward*.
4. Untuk mengestimasi *transition probability* dari *n* buah jalur yang berasal dari suatu *state*, gunakan persamaan 2.17 untuk menghitung dari masing-masing jalur tersebut untuk setiap kemungkinan jalur perpindahan antar-*state* pada langkah 2. Untuk mengestimasi *transition probability* dari sebuah jalur, gunakan persamaan 2.23 dengan pembilang berupa penjumlahandari jalur yang sama (yang sedang diestimasi) dan pembagi berupa penjumlahan dari *n* buah jalur tersebut untuk setiap kemungkinan jalur perpindahan antar-*state* pada langkah 2.
5. Untuk mengestimasi *emission probability* dari suatu *state* tertentu, gunakan persamaan 2.19 untuk menghitung $\gamma_t(i)$ dari setiap simbol yang dapat dihasilkan oleh *state* tersebut. Untuk mengestimasi *emission probability* dari sebuah simbol, gunakan persamaan 2.24 dengan pembilang berupa penjumlahan $\gamma_t(i)$ untuk simbol yang sama (yang sedang diestimasi) dan pembagi berupa penjumlahan seluruh $\gamma_t(i)$ dari *state* yang bersangkutan.
6. Memperbaharui *transition probability* untuk setiap jalur perpindahan dan *emission probability* untuk setiap *state* dari HMM dengan menggunakan estimasi *transition probability* dan *emission probability* yang diperoleh dari langkah 4 dan 5.
7. Melakukan iterasi langkah 2 sampai 6 sedemikian sehingga parameter-parameter dari HMM konvergen (berubah secara tidak signifikan).

2.3.2. Algoritma Viterbi

Algoritma Viterbi merupakan algoritma yang digunakan untuk memilih rangkaian *state* dalam suatu HMM yang dilalui oleh *sequence* observasi sedemikian sehingga probabilitasnya menjadi maksimal (Colton, 2007). Rumus yang digunakan pada algoritma viterbi telah diterangkan pada subbab 2.1.3. Dengan menggunakan algoritma Viterbi, dimungkinkan untuk mempertimbangkan seluruh *sequence* observasi secara langsung dan menemukan jalur perpindahan terbaik untuk *sequence* tersebut.

3. METODE PENELITIAN

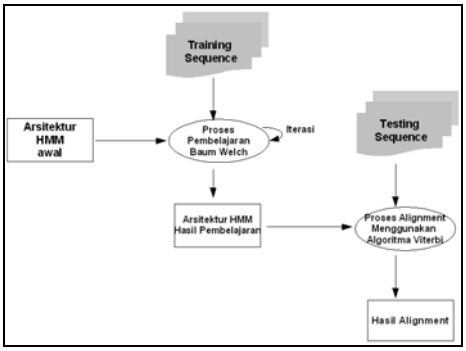
3.1. Analisis Sistem

Multiple Sequence Alignment terhadap kumpulan *sequence* protein merupakan langkah pertama yang diperlukan dalam bioinformatika untuk melakukan analisis lebih lanjut terhadap materi biologis seperti kesamaan homologi antar *sequence* dan analisis filogenetik untuk mengetahui asal hubungan evolusioner yang ada. *Multiple alignment* yang dilakukan pada kumpulan *sequence* protein lebih mudah dilakukan dibandingkan dengan *multiple alignment* pada *sequence* DNA. Pada *alignment sequence* DNA, harus diperhatikan pemilihan *reading frame*. *Alignment* yang benar dapat dihasilkan hanya jika pemilihan reading frame tepat. Permasalahan ini tidak dijumpai pada *alignment sequence* protein.

Sequence protein yang digunakan dalam *multiple sequence alignment* merupakan kumpulan string hasil *sequencing* dari sel mahluk hidup. Kumpulan string tersebut bermakna string asam amino atau protein. Pada penelitian ini, dibangun sebuah prototipe perangkat lunak untuk melakukan *multiple alignment* terhadap kumpulan *sequence* protein menggunakan metode HMM kemudian akan dilakukan pengujian untuk mengetahui

perfoma protipe perangkat lunak tersebut terhadap kumpulan sequence protein dengan kriteria – kriteria tertentu yang merepresentasikan permasalahan-permasalahan nyata yang sering dihadapi ketika melakukan alignment terhadap sequence protein. Untuk selanjutnya, prototipe perangkat lunak yang dibangun dalam penelitian ini disebut dengan ProHMMer.

HMM yang digunakan untuk melakukan *multiple alignment* terhadap *sequence* protein menggunakan HMM orde pertama. Proses pembelajaran dilakukan dengan menggunakan algoritma Baum-Welch terhadap himpunan *training sequence* secara berulang-ulang hingga parameter-parameter dalam HMM berubah tidak signifikan (stabil). Selanjutnya HMM hasil pembelajaran digunakan untuk melakukan *alignment* sekumpulan *test sequence,* menentukan *alignment sequence* dengan algoritma viterbi. Skema umum MSA menggunakan HMM yang digunakan dalam ProHMMer dapat dilihat pada Gambar 3.1.



Gambar 3.1 Skema Umum MSA dengan HMM

Sesuai dengan algoritma dalam metode *hidden markov model* seperti pada skema umum yang telah dijelaskan di atas, diperlukan suatu parameter *stopping condition* pada saat melakukan iterasi prosedur BaumWelch. Oleh karena itu, ProHMMer akan memberikan fasilitas berupa penentuan parameter *stopping condition*. Terdapat dua parameter yang dapat digunakan yaitu pertama banyaknya iterasi prosedur BaumWelch dan yang kedua, *maximum distance* yaitu selisih nilai hasil prosedur BaumWelch iterasi saat ini dikurangi nilai saat iterasi sebelumnya. Pengguna dapat memilih diantara kedua parameter tersebut, maksimum iterasi atau *maximum distance*.

Nilai BaumWelch merupakan nilai probabilitas model menghasilkan kumpulan *sequence*. Nilai BaumWelch didapatkan dengan melakukan rata – rata terhadap nilai probabilitas model menghasilkan *sequence*, yaitu rata – rata dari (rumus 2.5) dari semua *sequence* dalam kumpulan *sequence*. Secara komputasi, perkalian lebih mahal dibandingkan dengan penjumlahan (Gupta, 2004) sehingga ProHMMer menggunakan nilai logaritma berbasis e dari nilai probabilitas model menghasilkan *sequence*. Nilai BaumWelch pada ProHMMer merupakan nilai rata – rata terhadap nilai logaritma berbasis e dari nilai probabilitas model menghasilkan *sequence*. Sehingga apabila *maximum distance* diset dengan nilai 0,5 berarti dapat diartikan selisih nilai logaritma probabilitas model menghasilkan kumpulan *sequence* iterasi saat ini dikurangi nilai iterasi sebelumnya maksimal 0,5.

Secara umum, file yang digunakan untuk menyimpan kumpulan *sequence* berformat FASTA (*.tfa). Oleh karena itu, ProHMMer membutuhkan input berupa file berformat FASTA, berisi kumpulan *sequence* yang belum ter-*align*. ProHMMer juga mengeluarkan output berupa file berformat FASTA, berisi kumpulan *sequence* yang sudah ter-*align*.

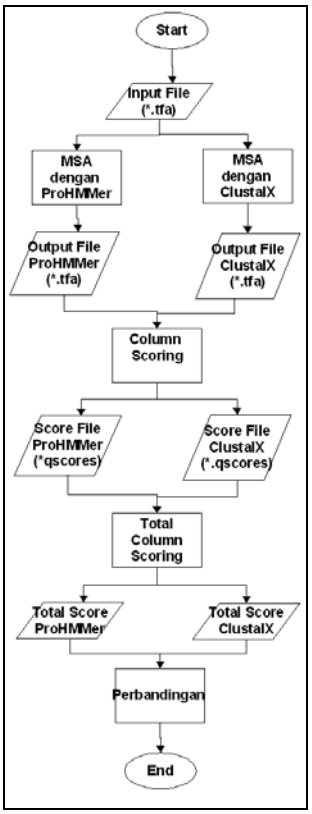
Perancangan sistem yang baik harus mempertimbangkan kemampuan komponen - komponen sistem tersebut untuk digunakan pada sistem yang lain. Analisis dan perancangan yang berorientasi objek memungkinkan hal tersebut. ProHMMer menggunakan analisis dan perancangan berorientasi objek. Salah satu tujuan yang ingin dicapai dalam pengembangan perangkat lunak berorientasi objek adalah reusability. ProHMMer menggunakan *class library* biojava. Terdapat beberapa bagian yang tidak dibutuhkan, maka *class library* ini kemudian dipaket ulang dengan hanya menyertakan bagian-bagian *class library* yang diperlukan bagi ProHMMer.

3.2. Rancangan Pengujian Performa ProHMMer

ProHMMer merupakan suatu sistem *multiple sequence alignment* yang menggunakan metode *hidden markov model*. Performa ProHMMer perlu diuji untuk menunjukkan bahwa *hidden markov model* merupakan suatu metode yang tidak buruk jika diterapkan dalam MSA, sebagaimana diketahui sebelumnya bahwa *hidden markov model* digunakan dalam proses *speech recognition* (Rabiner, 1989), serta untuk menunjukkan bahwa *hidden markov model* merupakan salah satu metode alternatif pengganti metode *dynamic programming* yang menghabiskan cukup banyak biaya komputasi dalam melakukan MSA.

Sebagai pembanding, untuk mengetahui perfoma ProHMMer terhadap kumpulan *sequence* protein dengan kriteria – kriteria tertentu yang merepresentasikan permasalahan-permasalahan nyata yang sering

dihadapi ketika melakukan *alignment* terhadap *sequence* protein maka hasil alignment ProHMMer akan dibandingkan dengan hasil alignment ClustalX. Menurut Thompson et al (1999), ClustalX merupakan program MSA yang sering digunakan oleh para praktisi biomolekuler, sehingga ClustalX dipilih sebagai pembanding untuk menentukan perfoma ProHMMer. Dalam melakukan *alignment*, ClustalX menggunakan metode komputasi *progressive pairwise alignment* yang merupakan perkembangan dari metode *dynamic programming*. Salah satu cara membandingkan hasil *alignment* untuk mengetahui kemampuan program dalam melakukan *alignment* secara tepat adalah dengan mencari nilai c*olumn score* dari hasil *alignment* tersebut (Thompson, 1999). Dalam tugas akhir ini, nilai *column score* didapatkan dengan menggunakan fasilitas Column Score pada ClustalX dengan parameter *protein weight matrix* Gonnet PAM 250. *Protein weight matrix* Gonnet PAM 250 merupakan parameter standar dalam ClustalX. *Score* untuk tiap *alignment* adalah penjumlahan dari *score* tiap kolomnya. ClustalX hanya menyediakan fasilitas Column Score, sehingga dibutuhkan sebuah *tool* untuk menghitung *score* total dari suatu *alignment,* dimana *score* total merupakan hasil penjumlahan dari *score* tiap kolomnya. Untuk mendapatkan *score* total, maka dibangun *tool* TotalColumnScoring.



Gambar 3.2. Diagram Alir Proses Pengujian ProHMMer

Data *sequence* protein yang digunakan untuk melakukan pengujian terhadap ProHMMer adalah BAliBASE (*Benchmark Alignment Database*) versi 3.0. Seperti dijelaskan pada sub bab 2.2.3.5, BaliBASE adalah sebuah dataset berisi *alignment sequence* protein yang khusus dibangun untuk membandingkan performa program MSA (Thompson et al., 1999). BAliBASE terdiri dari lima sub dataset dengan kriteria – kriteria tertentu yang merepresentasikan permasalahan-permasalahan nyata yang sering dihadapi ketika melakukan alignment terhadap *sequence* protein. Kriteria tersebut diantaranya panjang *sequence*, tingkat *identity* (kesamaan) diantara sequence, keberadaan *internal insertion* dan keberadaan N/C *terminal extension* di dalam *sequence* (Thompson, 1999). Terdapat 218 *file full length sequence* (BB) dan 168 *file truncated sequence* (BBS) yang terbagi menjadi lima sub dataset dalam BaliBASE versi 3.0. Karena keterbatasan *resource* komputasi yang dimiliki, maka untuk menjalankan pengujian setiap sub dataset diambil sampel lima file. Lima file tiap sub dataset diasumsikan telah cukup mewakili setiap sub dataset BAliBASE. Pengambilan sampel lima file dilakukan secara random terhadap seluruh file yang berada pada tiap sub dataset.

Diagram alir proses pengujian performa ProHMMer untuk mengetahui perfoma ProHMMer terhadap kumpulan *sequence* protein dengan kriteria – kriteria tertentu dengan membandingkan hasil alignment ProHMMer dengan ClustalX seperti dijelaskan sebelumnya dapat dilihat pada Gambar 3.2.

4. HASIL DAN PEMBAHASAN

4.1. Data, Parameter dan Prosedur Pengujian

Sesuai dengan rancangan pengujian sistem, untuk mengetahui perfoma ProHMMer terhadap kumpulan *sequence* protein dengan kriteria – kriteria tertentu yang merepresentasikan permasalahan-permasalahan nyata yang sering dihadapi ketika melakukan *alignment* terhadap *sequence* protein maka hasil alignment ProHMMer akan dibandingkan dengan hasil alignment ClustalX. Kriteria tersebut diantaranya panjang *sequence*, tingkat *identity* (kesamaan) diantara *sequence*, keberadaan *internal insertion* dan keberadaan N/C t*erminal extension* di dalam *sequence* (Thompson, 1999).

Seperti yang dijelaskan sebelumnya data *sequence* protein yang digunakan untuk melakukan pengujian terhadap ProHMMer adalah BAliBASE versi 3.0. Terdapat 218 *file full length sequence* (BB) dan 168 *file truncated sequence* (BBS) yang terbagi menjadi lima sub dataset dalam BaliBASE versi 3.0. Sub dataset dalam BaliBASE disebut juga *reference*, karena setiap subdataset mereferensikan kriteria *sequence* tertentu. Karena keterbatasan *resource* komputasi yang dimiliki, maka untuk menjalankan pengujian setiap sub dataset diambil sampel lima file. Lima file tiap sub dataset diasumsikan telah cukup mewakili setiap sub dataset BaliBASE, lima file BB dan lima file BBS yang sesuai. Pengambilan sampel lima file dilakukan secara random terhadap seluruh file yang berada pada tiap sub dataset.

Parameter *stopping criteria* ProHMMer yang akan digunakan dalam pengujian adalah *maximum distance*. ProHMMer akan diuji dengan nilai *maximum distance* 0,5. Sedangkan ClustalX menggunakan parameter standar, dengan asumsi parameter standar pada ClustalX adalah parameter yang terbaik. Sesuai dengan rancangan pengujian, setelah dihasilkan *file* output dari ProHMMer dan ClustalX, maka dilanjutkan dengan penilaian atau proses *scoring* dengan menggunakan fasilitas *column score* pada ClustalX. Setelah didapatkan hasil berupa file *column score* (*.qscores) dilanjutkan dengan proses perhitungan *score* total dengan menggunakan *tool* TotalColumnScoring.

4.2. Hasil Pengujian

ProHMMer menggunakan metode *Hidden Markov Model* dengan komponen utama berupa *state* dan *probability*, inisialisasi awal HMM dilakukan secara random, sehingga proses *alignment* ProHMMer terhadap suatu *file* tertentu dengan parameter yang sama menghasilkan *alignment* yang belum tentu sama persis. Oleh karena itu, pengujian ProHMMer dilakukan sebanyak 3 kali terhadap *file* dan parameter yang sama. *Score* yang digunakan untuk melakukan perbandingan dengan ClustalX adalah rata-rata *score* dari 3 kali pengujian tersebut. Hasil *score* menggunakan program ProHMMer (hasil rata – rata *score* tiga kali pengujian) dan ClustalX terangkum dalam Tabel 4.1.

Tabel 4.1. Score ProHMMer dan ClustalX

Reference	*File*	*Score*		
		ProHMMer 0.5	ClustalX	ProHMMer/ClustalX
11	BB11001.tfa	2356	2554	0.92
	BB11002.tfa	1140.67	1430	0.8
	BB11009.tfa	3161.33	2943	1.07
	BB11013.tfa	1212.33	1520	0.8
	BB11022.tfa	2064.67	2030	1.02
	BBS11001.tfa	2356	2411	0.98
	BBS11002.tfa	1345.33	1469	0.92
	BBS11009.tfa	3001.33	3038	0.99
	BBS11013.tfa	974.33	1304	0.75
	BBS11022.tfa	1581	1683	0.94
12	BB12003.tfa	1491.67	2044	0.73
	BB12006.tfa	8647.33	9004	0.96
	BB12009.tfa	1865.67	2697	0.69
	BB12014.tfa	1467.33	2367	0.62
	BB12025.tfa	5150.00	4499	1.14
	BBS12003.tfa	2081.67	2108	0.99
	BBS12006.tfa	8957	9002	0.99
	BBS12009.tfa	2623.33	2727	0.96
	BBS12014.tfa	1970	2025	0.97

Reference	*File*	Score		
		ProHMMer 0.5	ClustalX	ProHMMer/ClustalX
	BBS12025.tfa	3820	4052	0.94
20	BB20001.tfa	1495.67	2804	0.53
	BB20011.tfa	2891	3143	0.92
	BB20016.tfa	2159	3761	0.57
	BB20020.tfa	6751.33	8684	0.77
	BB20029.tfa	2643.67	2747	0.96
	BBS20001.tfa	2533	2493	1.02
	BBS20011.tfa	3274.67	3296	0.99
	BBS20016.tfa	3480.67	3412	1.02
	BBS20020.tfa	8583	8734	0.98
	BBS20029.tfa	2166.33	2141	1.01
30	BB30006.tfa	1842.33	2726	0.68
	BB30007.tfa	2175	2325	0.94
	BB30015.tfa	2006.67	2897	0.69
	BB30017.tfa	4725	5437	0.87
	BB30027.tfa	1896.67	2336	0.81
	BBS30006.tfa	1916	2389	0.8
	BBS30007.tfa	2684.67	2790	0.96
	BBS30015.tfa	2254	2465	0.91
	BBS30017.tfa	4877.67	5437	0.89
	BBS30027.tfa	2139.67	2284	0.94
40	BB40006.tfa	7698.33	9157	0.84
	BB40010.tfa	2148.67	2887	0.74
	BB40014.tfa	6260.33	10757	0.58
	BB40018.tfa	2107.67	3671	0.57
	BB40045.tfa	2513.67	2685	0.94
50	BB50002.tfa	4488.33	5173	0.87
	BB50005.tfa	26209.33	29230	0.89
	BB50007.tfa	3229	3981	0.81
	BB50008.tfa	5034.67	7533	0.66
	BB50013.tfa	6508	8622	0.75
	BBS50002.tfa	3038	3410	0.89
	BBS50005.tfa	25032.67	28775	0.87
	BBS50007.tfa	3735.67	3846	0.97
	BBS50008.tfa	4458.33	5225	0.85
	BBS50013.tfa	5359	7009	0.77

Dari Tabel 4.2 dapat dilihat bahwa performa ProHMMer berbanding ClustalX meningkat pada penggunaan *truncated sequence*. Performa ProHMMer cukup baik pada *Reference* 1, Reference 2, *Reference* 3 kemudian turun pada *Reference* 4, dan agak naik pada *Reference* 5.

HMM digunakan dalam MSA sebagai alternatif untuk menghasilkan MSA yang bersifat prediktif dan tidak terikat pada satu *scoring matrix* tertentu. Harapannya, dengan penggunaan HMM, bisa didapatkan *alignment* yang dapat memprediksi *aligned region* yang baru dari suatu kumpulan *sequence*. HMM hanya berusaha menyusun simbol-simbol pada *sequence* sedemikian sehingga susunannya

mencerminkan pola yang didapat pada saat pelatihan. Apabila *sequence* yang digunakan pada pelatihan banyak mengandung kesamaan (*identity* tinggi) maka pola yang ditangkap oleh model lebih jelas sehingga kualitas *alignment* yang dihasilkan juga semakin baik. Hal ini dapat dilihat pada data pengujian Tabel 4.3. yang menunjukkan bahwa ProHMMer memiliki performa cukup baik pada data yang memiliki tingkat *identity* tinggi yaitu *Reference* 1, 2, dan 3.

Pada data yang mengandung *noise* yaitu *N/C terminal extensions* atau *insertion*, ProHMMer mengalami penurunan perfoma. Hal ini dapat dilihat dari Tabel 4.2. , ProHMMer mengalami penurunan pada *Reference* 4 dan *Reference* 5, walaupun pada *Reference* 5, ProHMMer mengalami kenaikan sedikit. Performa ProHMMer mengalami penurunan pada data dengan *sequence* yang panjang, hal ini bisa dilihat ketika digunakan pada *truncated sequence* dibandingkan dengan ketika digunakan pada *full length sequence.* Performa ProHMMer meningkat cukup signifikan ketika digunakan pada *truncated sequence.* Hal ini disebabkan *truncated sequence* mengandung area terkonservasi yang memiliki *identity* lebih tinggi, memiliki panjang lebih pendek dan juga jumlah noise lebih sedikit daripada *full length sequence*.

5. KESIMPULAN

Berdasarkan hasil penelitian, analisis dan pembangunan sistem *multiple sequence alignment* menggunakan *hidden markov model* yaitu ProHMMer dan pengujian ProHMMer , maka dapat ditarik beberapa kesimpulan sebagai berikut :

1. *Hidden Markov Model* dapat menjadi alternatif metode untuk menghasilkan *multiple sequence alignment* yang bersifat prediktif dan tidak terikat pada satu *scoring matrix* tertentu.
2. Algoritma Baum-Welch dapat digunakan untuk mengestimasi parameter-parameter dalam HMM.
3. Algoritma Viterbi dapat diterapkan untuk menghasilkan *alignment* dari *unaligned sequence*.
4. Implementasi HMM pada ProHMMer memiliki performa lebih baik pada data *sequence* yang memiliki *identity* tinggi dan mengalami penurunan perfoma pada data *sequence* yang panjang dan data sequence yang memiliki banyak *noise* seperti N/C *terminal extension* atau *insertion*.

6. DAFTAR PUSTAKA

Birney, E., 2001, *Hidden Markov Models in Biological Sequence Analysis,* Volume 45, IBM Journal of Research and Development.

Booch,G., Rumbaugh,J., Jacobson,I., 2005*, The Unified Modeling Language User Guide.* Addison Wesley Professional.

Colton, S., 2007, *Introduction to Bioinformatics, Genetics Background, Course 3 41 Lecture Slide.* Department of Computing Imperial College, London.

Gupta, S., 2004, *Hardware Acceleration of Hidden Markov Models for Bioinformatics Applications,* Boise State University.

Hughey, R., Krogh, A. , 1996, *Hidden Markov Models for Sequence Analysis : Extension and Analysis of The Basic Method,* University of California, Santa Cruz.

Koski, T., 2001, *Hidden Markov Models for Bioinformatics,* Kluwer Academic Publishers,Netherlands.

Krogh, A., Brown, M., Mian, I.S., Sjölander, K., Haussler, D., 1992*, Hidden Markov Models in Computational Biology : Application to Protein Modeling,* University of California, Santa Cruz.

Rabiner, L.R., 1989, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,* Proceedings of the IEEE, Vol. 77, No. 2, pp. 257-286.

Schmidt,J.W., Matthes,F.,Niederée,C., 1999, *Object-Oriented Analysis and Design Course Lecture Slide,* TU Hamburg, Harburg.

Thompson, J.D., Frederic, P., Olivier, P., 1999, *A Comprehensive Comparison of Multiple Sequence Alignment Programs, Nucleic Acid Research*, Vol. 27, No. 13

Xiong, J., 2006. *Essential Bioinformatics.,* Cambridge University Press, Cambridge.