

Penerapan Algoritma Branch and Bound pada HP Model untuk Penyelesaian Protein Folding Problem

Yavie Azka Putra Araly - 13524077

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jalan Ganesha 10 Bandung

E-mail: yavieazkaputra@gmail.com , 13524077@std.stei.itb.ac.id

Abstrak—*Protein folding problem* merupakan salah satu masalah yang masih belum terpecahkan di bidang bioinformatika komputasional yang bertujuan untuk menentukan struktur tiga dimensi dari suatu sekuens asam amino protein dengan energi minimum. Permasalahan ini sudah terbukti tergolong *NP-Hard*, bahkan dengan model yang disederhanakan sekalipun, sehingga membutuhkan pendekatan algoritmik yang efisien untuk dapat diselesaikan. Makalah ini mengkaji penerapan algoritma *Branch and Bound* untuk menyelesaikan permasalahan protein folding pada model *Hydrophobic-Polar* (HP) pada dua dimensi. Model ini merupakan abstraksi yang menyederhanakan asam amino menjadi dua tipe berdasarkan sifat hidrofobitasnya. Makalah ini juga menampilkan hasil pengujian implementasi pada enam sekuens asam amino yang membuktikan efektivitas mekanisme *pruning* yang diterapkan pada algoritma *Branch and Bound* meskipun kompleksitas kasus terburuk tetap eksponensial.

Kata Kunci—*Branch and Bound; protein folding; HP Model; bioinformatika; NP-Hard*

I. PENDAHULUAN

Protein merupakan molekul biologis fundamental yang menjalankan hampir seluruh fungsi seluler dalam tubuh makhluk hidup, mulai dari katalisis reaksi biokimia, pengangkutan molekul, hingga pertahanan imunitas. Fungsi protein ditentukan oleh struktur tiga dimensi (3D) yang terbentuk melalui proses yang dikenal sebagai *protein folding*, yaitu pelipatan rantai asam amino linear menjadi konformasi 3D yang stabil. Anfinsen [1] membuktikan bahwa struktur 3D ini sepenuhnya ditentukan oleh sekuens asam amino, sehingga prediksi struktur protein dari sekuensnya menjadi salah satu permasalahan sentral dalam bioinformatika komputasional.

Kegagalan proses *folding* yang dikenal sebagai *protein misfolding* terbukti menjadi pangkal berbagai penyakit degeneratif serius. Akumulasi protein *misfolded* dikaitkan dengan penyakit Alzheimer, Parkinson, dan *Amyotrophic Lateral Sclerosis* (ALS) [2]. Pemahaman mendalam terhadap mekanisme *folding* memiliki implikasi langsung dalam pengembangan terapi dan perancangan obat (*drug design*) berbasis struktur protein.

Secara komputasional, prediksi struktur protein menghadapi tantangan yang disebut Levinthal's Paradox [3]. meskipun ruang konformasi protein berorde 10^{300} kemungkinan, protein biologis mampu mencapai konformasi

stabil dalam hitungan milidetik. Hal ini menunjukkan bahwa proses folding tidak bersifat *exhaustive search*, melainkan mengikuti jalur tertentu yang diarahkan oleh hukum fisikokimia, terutama hydrophobic effect. Dari sudut pandang komputasi, permasalahan ini terbukti bersifat NP-Hard bahkan pada model yang disederhanakan sekalipun [4], sehingga tidak ada algoritma exact berorde polinomial yang diketahui hingga saat ini.

Berbagai pendekatan telah dikembangkan untuk mengatasi kompleksitas ini. Pendekatan berbasis *machine learning*, khususnya AlphaFold2 yang dikembangkan oleh Google DeepMind [5], telah merevolusi prediksi struktur protein dengan tingkat akurasi yang mendekati hasil eksperimental. Meskipun demikian, pendekatan berbasis *learning* bersifat aproksimasi ini tidak memberikan jaminan optimalitas secara matematis dan sangat bergantung pada ketersediaan data protein homolog. Dalam konteks *drug design* dan validasi model, algoritma *exact* yang menjamin solusi optimal tetap memiliki peran yang tidak tergantikan.

Untuk memungkinkan studi komputasional yang traktabel, Lau dan Dill [6] memperkenalkan *Hydrophobic-Polar* (HP) *Model*, yaitu sebuah abstraksi yang menyederhanakan asam amino menjadi dua tipe: H (*hydrophobic*) dan P (*polar*), dengan protein dimodelkan sebagai rantai pada kisi (*lattice*) dua dimensi. Meskipun merupakan simplifikasi, HP *Model* terbukti NP-Complete [4] dan mampu menangkap *driving force* utama *folding* secara biologis, yakni kecenderungan residu hidrofobik untuk berkumpul di inti protein guna menghindari kontak dengan pelarut air. Minimasi energi pada HP *Model* dirumuskan sebagai maksimasi kontak H-H yang tidak terhubung langsung pada rantai (*topological contacts*).

Dalam makalah ini, diusulkan penerapan algoritma *Branch and Bound* (BnB) untuk menyelesaikan *protein folding* pada HP *Model* secara *exact*. Algoritma BnB dipilih karena kemampuannya mereduksi ruang pencarian secara signifikan melalui mekanisme *pruning* berbasis fungsi batas (*bounding function*), sehingga menjamin ditemukannya solusi optimal tanpa harus mengeksplorasi seluruh ruang konformasi. Pendekatan ini mengikuti formulasi BnB sebagai *least cost search* pada pohon ruang status (*state space tree*) [7], di mana

setiap simpul merepresentasikan konformasi parsial protein pada kisi 2D.

Kontribusi utama makalah ini adalah:

1. Pemodelan *protein folding* pada HP Model 2D sebagai permasalahan *exact optimization* menggunakan kerangka BnB dengan fungsi $cost \hat{c}(i) = \hat{f}(i) + \hat{g}(i)$.
2. Perancangan fungsi *upper bound* $\hat{g}(i)$ berbasis pasangan H-H *eligible* yang tersisa untuk memaksimalkan efisiensi *pruning*.
3. Penerapan eliminasi simetri untuk mereduksi ruang pencarian hingga seperdelapan dari ruang pencarian penuh.
4. Evaluasi empiris efektivitas algoritma terhadap sekuens *benchmark* standar HP Model.

II. LANDASAN TEORI

A. Protein dan Struktur Biologisnya

Protein adalah molekul yang sangat penting dalam biologi. Di dalam sel tubuh manusia, terdapat 18,000 tipe protein yang memiliki fungsi berbeda-beda. Protein tersusun dari rantai asam amino yang dihubungkan oleh ikatan peptida. Terdapat 20 jenis asam amino standar yang menjadi penyusun protein, masing-masing dibedakan oleh rantai samping (*side chain*) dengan sifat fisikokimia yang berbeda-beda, meliputi sifat hidrofobik, polar, bermuatan positif, maupun bermuatan negatif [8]. Urutan linier asam amino dalam rantai protein disebut sebagai *primary structure* atau struktur primer, yang secara genetis dikodekan oleh urutan basa nitrogen pada DNA.

Struktur protein tidak berhenti pada tataran linier. Dalam lingkungan seluler, rantai asam amino mengalami pelipatan spontan membentuk arsitektur tiga dimensi yang hierarkis. Hierarki ini umumnya diklasifikasikan ke dalam empat level struktur [2], yaitu *primary structure*, *secondary structure*, *tertiary structure*, dan *quaternary structure*.

Di antara keempat level tersebut, struktur tersier memiliki relevansi komputasional paling tinggi dalam konteks *protein folding problem*, karena menjadi target prediksi yang langsung berimplikasi pada fungsi biologis. Kegagalan protein mencapai struktur tersier yang benar yang dikenal sebagai *protein misfolding* berkorelasi kuat dengan sejumlah penyakit degeneratif. Akumulasi protein *misfolded* dalam bentuk agregat fibrilar yang tidak larut telah dikaitkan secara mekanistik dengan penyakit Alzheimer melalui deposisi peptida amiloid-beta, penyakit Parkinson melalui pembentukan *Lewy bodies* dari protein alfa-sinuklein, serta *Amyotrophic Lateral Sclerosis* (ALS) melalui agregasi protein SOD1 [9]. Pemahaman komputasional terhadap proses *folding* karenanya tidak hanya bernilai akademis, melainkan membuka peluang intervensi terapeutik melalui perancangan molekul yang dapat menstabilkan atau memodulasi konformasi protein target.

B. Protein Folding Problem

Protein folding problem secara formal didefinisikan sebagai berikut: diberikan sekuens asam amino suatu protein, tentukan konformasi tiga dimensi dengan energi bebas (*free energy*)

minimum yang dapat dicapai oleh protein tersebut [10]. Konformasi dengan energi minimum ini disebut sebagai *native state*, yaitu struktur stabil yang secara biologis fungsional. Landasan teoritis problem ini diletakkan oleh Anfinsen [1] melalui eksperimen denaturasi dan renaturasi ribonuklease A, yang membuktikan bahwa *native state* protein sepenuhnya dan secara unik ditentukan oleh sekuens asam aminonya, tanpa memerlukan informasi tambahan dari luar. Temuan ini, yang kemudian dikenal sebagai *Anfinsen's Dogma*, menegaskan bahwa *protein folding problem* pada dasarnya adalah permasalahan optimasi, yaitu pencarian konformasi dengan energi terendah dalam ruang konformasi yang sangat besar.

C. HP Model

Mengingat kompleksitas model *all-atom* yang melibatkan ruang konformasi kontinu dan berdimensi tinggi, dibutuhkan suatu abstraksi yang menyederhanakan permasalahan tanpa menghilangkan esensi komputasionalnya. Lau dan Dill [6] memperkenalkan *Hydrophobic-Polar* (HP) Model. Model ini merupakan sebuah model *lattice* yang menyederhanakan protein sedemikian rupa sehingga tetap dapat dianalisis secara komputasional, sambil mempertahankan *hydrophobic effect* sebagai *driving force* utama *folding* [11].

Dalam HP Model, setiap asam amino diklasifikasikan ke dalam salah satu dari dua kategori berdasarkan sifat kelarutannya dalam air. H (*Hydrophobic*), merepresentasikan residu nonpolar yang cenderung menghindari kontak dengan pelarut air, dan P (*Polar*), merepresentasikan residu polar atau bermuatan yang cenderung berinteraksi baik dengan air. Suatu protein dengan panjang n direpresentasikan sebagai string biner $S = s_1 s_2 \dots s_n$, dengan $s_i \in \{H, P\}$ untuk setiap $i = 1, 2, \dots, n$.

Konformasi protein dimodelkan sebagai *embedding* dari string S ke dalam *lattice* (kisi) diskret yang pada umumnya berupa kisi persegi dua dimensi $\mathbb{Z}^2$, meskipun ekstensi ke kisi tiga dimensi $\mathbb{Z}^3$ juga umum digunakan dalam literatur [12]. Setiap asam amino s_i ditempatkan pada satu titik koordinat $(x_i, y_i) \in \mathbb{Z}^2$, dengan dua batasan utama yang harus dipenuhi:

1. Tidak terdapat dua asam amino yang menempati titik koordinat yang sama, atau $(x_i, y_i) \neq (x_j, y_j)$ untuk setiap $i \neq j$.
2. Asam amino yang bertetangga dalam sekuens harus bertetangga pula pada kisi, atau $|x_i - x_{i+1}| + |y_i - y_{i+1}| = 1$ untuk setiap $i = 1, \dots, n-1$.

Energi suatu konformasi pada HP Model didefinisikan berdasarkan jumlah kontak topologis antara residu H. Suatu pasangan (i, j) dengan $i < j$ dikatakan membentuk *topological contact* jika memenuhi dua syarat. Pertama, kedua residu bersifat hidrofobik ($s_i = s_j = H$), dan kedua residu bertetangga pada kisi namun tidak terhubung langsung pada rantai sekuens ($|x_i - x_j| + |y_i - y_j| = 1$ dan $|i - j| > 1$) [6]. Syarat $|i - j| > 1$ ini penting karena residu yang bertetangga langsung dalam rantai akan selalu bertetangga pula pada kisi sebagai konsekuensi dari batasan kedua, sehingga tidak memberikan informasi mengenai kualitas pelipatan.

Energi total konformasi dirumuskan sebagai:

$$E = - \sum_{i,j} \delta_{ij}$$

dengan $\delta_{ij} = 1$ jika pasangan (i, j) membentuk *topological contact*, dan $\delta_{ij} = 0$ untuk kondisi lainnya. Tanda negatif pada formulasi ini merefleksikan prinsip fisika bahwa pembentukan kontak hidrofobik menurunkan energi bebas sistem, dimana setiap residu H yang berhasil bersembunyi dari pelarut air melalui kontak dengan residu H lain akan mengurangi energi permukaan yang tidak diinginkan secara termodinamika [6]. Konsekuensinya, pencarian konformasi dengan energi minimum ($\min E$) secara matematis ekuivalen dengan pencarian konformasi yang memaksimalkan jumlah *topological contact* H-H.

Meskipun HP Model merupakan simplifikasi yang signifikan dari kompleksitas kimiawi protein yang sesungguhnya, yaitu mereduksi 20 jenis asam amino menjadi hanya dua kategori dan mengabaikan ikatan hidrogen maupun interaksi elektrostatis, model ini tetap relevan secara biologis karena menangkap mekanisme yang oleh Dill [6] diidentifikasi sebagai kontributor termodinamika terbesar dalam proses *folding*, yakni efek hidrofobik. Validitas pendekatan ini didukung oleh pengamatan bahwa pola distribusi residu hidrofobik pada protein nyata cenderung terkonsentrasi di bagian inti struktur, sementara residu polar terdistribusi di permukaan yang bersentuhan dengan pelarut [13].

Berger dan Leighton [4] membuktikan bahwa *protein folding problem* pada HP Model dengan kisi kubik tiga dimensi bersifat NP-Complete. Hasil ini kemudian dilengkapi oleh Crescenzi *et al.* [17] yang membuktikan kompleksitas serupa untuk kasus kisi dua dimensi. Implikasi dari kedua hasil ini adalah bahwa meskipun HP Model secara signifikan menyederhanakan permasalahan *protein folding* dibandingkan model *all-atom*, kompleksitas komputasionalnya tetap setara dengan permasalahan optimasi kombinatorial NP-Hard lainnya, seperti *Traveling Salesman Problem* dan *Knapsack Problem*. Hal ini menjustifikasi penggunaan HP Model sebagai *benchmark* yang bermakna secara teoritis untuk menguji efektivitas algoritma *exact* seperti *Branch and Bound*, karena keberhasilan algoritma pada model ini mencerminkan kemampuannya menangani permasalahan NP-Hard pada umumnya.

Salah satu karakteristik penting HP Model yang relevan bagi efisiensi algoritma pencarian adalah keberadaan simetri pada *lattice*. Setiap konformasi valid memiliki delapan representasi yang ekuivalen secara energetik, yang dihasilkan dari empat rotasi ($0^\circ, 90^\circ, 180^\circ, 270^\circ$) dikombinasikan dengan refleksi cermin [14]. Tanpa penanganan khusus, algoritma pencarian akan mengeksplorasi kedelapan representasi ini secara redundan meskipun merepresentasikan solusi yang identik secara struktural. Eliminasi simetri ini dapat mereduksi ruang pencarian efektif hingga seperdelapan dari ruang pencarian penuh tanpa mengurangi kelengkapan pencarian solusi optimal.

D. Algoritma Branch and Bound

Algoritma *Branch and Bound* (BnB) merupakan algoritma yang digunakan untuk persoalan optimasi, dimana persoalan ini memiliki objektif untuk minimasi/maksimasi suatu fungsi objektif yang tidak melanggar batasan persoalan terkait [7]. BnB menggunakan pendekatan yang mirip dengan BFS (*Breadth First Search*), namun dengan kombinasi *least cost search*, dimana simpul yang di-*expand* selanjutnya tidak lagi berdasarkan FIFO seperti di BFS, melainkan berdasarkan *cost* terkecil (pada kasus minimasi) atau *cost* terbesar (pada kasus maksimasi). Berbeda dengan *Backtracking* yang menggunakan *depth-first order* dalam pembangkitan simpulnya, BnB membangkitkan simpul dengan beberapa ‘aturan’ tertentu, dimana salah satu aturan paling umum adalah ‘*best-first rule*’, dimana aturan ini akan memilih untuk membangkitkan simpul terbaik dari seluruh simpul hidup.

Setiap instansiasi BnB memiliki 4 komponen utama yang saling berinteraksi:

1. State Space Tree

State space tree merupakan struktur pohon yang merepresentasikan seluruh ruang pencarian. Setiap simpul (*node*) dalam pohon merepresentasikan suatu state parsial dari solusi yang sedang dibangun. Simpul akar (*root*) merepresentasikan state awal sebelum keputusan apapun diambil, simpul internal merepresentasikan solusi parsial, dan simpul daun (*leaf*) merepresentasikan solusi lengkap atau state buntu yang tidak dapat dikembangkan lebih lanjut.

2. Fungsi cost

Fungsi *cost* $\hat{c}(i)$ merupakan estimasi ongkos total untuk mencapai solusi optimal melalui simpul i . Fungsi ini didefinisikan sebagai:

$$\hat{c}(i) = \hat{f}(i) + \hat{g}(i)$$

di mana $\hat{f}(i)$ adalah ongkos yang telah dikeluarkan untuk mencapai simpul i dari simpul akar, dan $\hat{g}(i)$ adalah estimasi heuristik ongkos terkecil dari simpul i menuju simpul solusi pada subpohon yang berakar di i . Karena letak simpul solusi pada umumnya tidak diketahui, $\hat{g}(i)$ dihitung secara heuristik sebagai *lower bound* dari ongkos tersisa [7].

3. Priority Queue

Priority queue merupakan struktur data yang menyimpan semua simpul hidup, yaitu simpul yang telah dibangkitkan namun belum diekspansi. Simpul dengan nilai $\hat{c}(i)$ terkecil selalu dipilih sebagai simpul berikutnya yang akan diekspansi (*E-node*), sehingga menjamin bahwa pencarian selalu difokuskan pada cabang yang paling menjanjikan.

4. Fungsi Pembatas (Bounding Function)

Fungsi pembatas (*bounding function*) merupakan mekanisme yang menentukan apakah suatu simpul layak untuk diekspansi lebih lanjut. Terdapat beberapa kriteria fungsi pembatas. Pertama, simpul dipangkas jika nilai $\hat{c}(i)$ tidak lebih baik dari nilai solusi terbaik yang ditemukan sejauh ini (*best solution so far*). Kedua, simpul dipangkas jika tidak merepresentasikan solusi yang *feasible* karena melanggar batasan (*constraint*)

permasalahan. Ketiga, jika simpul merupakan simpul daun, nilai fungsi objektifnya dibandingkan dengan solusi terbaik saat ini dan yang terbaik dipertahankan.

Prosedur BnB secara lengkap dilakukan sebagai berikut. Pertama, simpul akar dimasukkan ke dalam *priority queue* Q . Jika simpul akar merupakan simpul solusi, pencarian selesai. Kedua, jika Q kosong, algoritma berhenti dan tidak ada solusi yang ditemukan. Ketiga, jika Q tidak kosong, dipilih simpul i dengan nilai $\hat{c}(i)$ terkecil dari Q sebagai *E-node*. Keempat, jika simpul i adalah simpul solusi, solusi optimal telah ditemukan. Pada permasalahan optimasi, seluruh simpul hidup dengan $\hat{c}$ lebih besar dari nilai solusi ini kemudian dimatikan. Kelima, jika simpul i bukan simpul solusi, seluruh anaknya dibangkitkan dan untuk setiap anak j , dihitung $\hat{c}(j)$ dan dimasukkan ke dalam Q . Prosedur kembali ke langkah kedua.

Kompleksitas BnB sangat bergantung pada kualitas fungsi pembatas $\hat{g}(i)$ yang digunakan. Pada kasus terburuk, di mana fungsi batas tidak mampu memangkas cabang apapun, BnB berperilaku identik dengan *exhaustive search* dengan kompleksitas eksponensial. Namun pada kasus rata-rata dengan fungsi batas yang baik, jumlah simpul yang dieksplorasi dapat berkurang secara drastis [15]. Untuk permasalahan NP-Hard seperti HP Model, kompleksitas *worst case* BnB adalah $O(3^n)$ yang mengikuti *branching factor* maksimum tiga pada kisi 2D, namun secara empiris jumlah simpul yang dieksplorasi jauh lebih kecil berkat efektivitas *pruning* [4].

III. PENYELESAIAN MASALAH

A. Definisi Formal Permasalahan

Diberikan sekuens asam amino $S = s_1 s_2 \dots s_n$ dengan $s_i \in \{H, P\}$, permasalahan yang diselesaikan dalam makalah ini adalah menentukan *embedding* dari S ke dalam kisi 2D $\mathbb{Z}^2$ yang memaksimalkan jumlah kontak topologis H-H. Secara formal, dicari fungsi penempatan $\phi: \{1, 2, \dots, n\} \rightarrow \mathbb{Z}^2$ yang memenuhi:

1. $\phi(i) \neq \phi(j)$ untuk setiap $i \neq j$
2. $\|\phi(i) - \phi(i+1)\|_1 = 1$ untuk setiap $i = 1, \dots, n-1$
3. $E^* = \sum_{i,j} \delta_{ij}$ bernilai maksimal

dengan $\delta_{ij} = 1$ jika $s_i = s_j = H$, $\|\phi(i) - \phi(j)\|_1 = 1$, dan $|i - j| > 1$.

B. Penerapan Algoritma Branch and Bound pada HP Model

Dalam penyelesaian masalah pada HP model dengan algoritma BnB, perlu diinstansiasikan berbagai komponen dalam BnB terlebih dahulu. Seperti yang sudah dijelaskan pada bagian sebelumnya, komponen ini terdiri dari representasi *state*, fungsi *cost*, fungsi *upper bound*, dan kriteria *pruning*.

Setiap simpul pada *state space tree* merepresentasikan suatu konformasi parsial protein, yakni penempatan k asam amino pertama ($s_1, s_2, \dots, s_k$) pada kisi 2D, dengan $1 \leq k \leq n$. *State* pada setiap simpul didefinisikan oleh tiga komponen, yaitu himpunan koordinat asam amino yang telah ditempatkan $\{(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k)\}$, jumlah kontak H-H yang telah

terbentuk (*contacts_so_far*), dan indeks asam amino berikutnya yang akan ditempatkan ($k+1$).

Simpul akar merepresentasikan penempatan s_1 pada titik asal (0,0). Ekspansi dari setiap simpul dilakukan dengan mencoba menempatkan asam amino berikutnya pada salah satu sel tetangga yang valid, yaitu sel yang belum ditempati dan dapat dicapai dari posisi asam amino terakhir. Pada kisi 2D dengan konektivitas berbasis *Manhattan distance*, terdapat maksimal tiga arah valid dari setiap posisi (mengecualikan arah yang datang untuk menghindari *self-intersection* trivial), sehingga *branching factor* maksimum adalah tiga [16]. Simpul daun tercapai ketika seluruh n asam amino telah ditempatkan, atau ketika tidak terdapat sel tetangga yang valid (*dead end*).

Karena HP Model merupakan permasalahan maksimasi (memaksimalkan jumlah kontak H-H), fungsi *cost* BnB diadaptasi dengan menerapkan negasi sehingga permasalahan dikonversi menjadi minimasi. Fungsi *cost* untuk simpul i yang merepresentasikan konformasi parsial pada level k didefinisikan sebagai:

$$\hat{c}(i) = \hat{f}(i) + \hat{g}(i)$$

dengan:

$$\begin{aligned}\hat{f}(i) &= -\text{contacts_so_far}(i) \\ \hat{g}(i) &= -\text{upper_bound_remaining}(i)\end{aligned}$$

sehingga:

$$\hat{c}(i) = -(\text{contacts_so_far}(i) + \text{upper_bound_remaining}(i))$$

Simpul dengan nilai $\hat{c}(i)$ terkecil (paling negatif) merepresentasikan konformasi parsial yang paling menjanjikan dan diprioritaskan untuk diekspansi lebih dahulu.

Kualitas fungsi *upper bound* secara langsung menentukan efektivitas *pruning*, termasuk efisiensi keseluruhan algoritma. Untuk simpul i pada level k , *upper bound* kontak yang masih mungkin terbentuk dari asam amino s_{k+1} hingga s_n dihitung sebagai jumlah pasangan H-H yang masih *eligible* untuk membentuk kontak:

$$\text{upper_bound_remaining}(i) = |\{(a, b): s_a = s_b = H, |a - b| > 1, a \geq k+1 \text{ atau } b \geq k+1\}|$$

Estimasi ini merupakan *upper bound* yang valid karena setiap pasangan H-H *eligible* yang tersisa paling banyak dapat membentuk satu kontak pada kisi, namun tidak semua pasangan dapat merealisasikan kontak secara simultan akibat keterbatasan geometri kisi. Sifat *upper bound* ini menjamin bahwa nilai estimasi tidak pernah lebih kecil dari jumlah kontak aktual yang dapat terbentuk, sehingga *pruning* yang dihasilkan tetap menjaga kelengkapan pencarian.

Terdapat tiga kriteria *pruning* yang diterapkan pada instansiasi HP Model ini. Kriteria pertama adalah *bound-based pruning*, dimana simpul i dipangkas jika nilai *upper bound* totalnya tidak melampaui solusi terbaik yang telah ditemukan,

$$\text{contacts_so_far}(i) + \text{upper_bound_remaining}(i) \leq \text{best_contacts_found}$$

Kondisi ini menjamin bahwa tidak ada solusi yang lebih baik dari *best_contacts_found* yang dapat ditemukan melalui simpul *i*, sehingga seluruh subpohon di bawah simpul *i* dapat dipangkas dengan aman.

Kriteria kedua adalah *infeasibility pruning*, dimana simpul dipangkas jika posisi yang hendak ditempati oleh asam amino berikutnya sudah terisi oleh asam amino lain, melanggar batasan pertama pada definisi permasalahan. Kondisi ini terdeteksi secara langsung melalui pengecekan terhadap himpunan sel yang telah ditempati.

Kriteria ketiga adalah *dead end pruning*, dimana simpul dipangkas jika tidak terdapat satupun arah gerak yang valid dari posisi asam amino terakhir, yaitu seluruh sel tetangga telah terisi. Kondisi ini merepresentasikan konformasi parsial yang tidak dapat dikembangkan lebih lanjut menjadi solusi lengkap.

Selain itu, sebagaimana dibahas pada Bagian II.C, setiap konformasi valid pada kisi 2D memiliki delapan representasi ekuivalen. Untuk mengeliminasi redundansi ini, diterapkan tiga aturan fiksasi pada tahap inisialisasi. Asam amino pertama s_1 selalu ditempatkan pada titik asal (0,0). Asam amino kedua s_2 selalu ditempatkan pada (1,0) (arah *right*), dan belokan pertama dari s_2 ke s_3 hanya diperbolehkan ke arah *up* atau *straight* (ke kanan), mengeliminasi refleksi cermin. Ketiga aturan ini mereduksi ruang pencarian efektif menjadi seperdelapan dari ruang pencarian penuh tanpa mengurangi kelengkapan, dimana setiap kelas ekuivalensi simetri direpresentasikan tepat oleh satu konformasi yang dieksplorasi.

C. Desain Algoritma

Sebelum pencarian dimulai, diterapkan tiga aturan fiksasi untuk mengeliminasi simetri rotasi dan refleksi pada kisi 2D:

1. s_1 ditempatkan pada (0, 0)
2. s_2 ditempatkan pada (1, 0) (arah RIGHT)
3. Dari s_2 , arah pertama hanya UP atau RIGHT untuk mengeliminasi refleksi cermin

Kemudian, setiap simpul pada *state space tree* direpresentasikan oleh struktur berikut:

```
State {
    placed : array koordinat ( $x_i, y_i$ ) untuk setiap AA
              yang sudah ditempatkan
    occupied : set koordinat yang sudah terisi (untuk
              pengecekan  $O(1)$ )
    level : indeks AA berikutnya ( $k+1$ )
    contacts : jumlah H-H contacts sejauh ini
}
```

Dengan menggunakan fungsi $\text{cost } \hat{c}(i)$ dan $\text{upper_bound_remaining}$ yang telah dijelaskan pada bagian sebelumnya, pseudocode lengkap dari algoritma ini adalah sebagai berikut:

Input : Sekuens $S = s_1 s_2 \dots s_n$

Output: Konformasi optimal dan nilai E^*

```
1. Inisialisasi:
   placed ← {(0,0), (1,0)}
   occupied ← {(0,0), (1,0)}
   contacts ← countContacts( $s_1, s_2, \text{placed}$ )
   best ← 0
   best_cfg ← {}

2. Masukkan state awal ke priority queue Q
   dengan  $\hat{c} = -(\text{contacts} + \text{UB\_remaining})$ 

3. while Q tidak kosong:
4.   i ← ExtractMin(Q) // simpul E-node
5.   // Pruning kriteria 1
6.   if i.contacts + UB_remaining(i) ≤ best:
7.     continue // PRUNE
8.   // Base case: solusi lengkap
9.   if i.level == n+1:
10.    if i.contacts > best:
11.      best ← i.contacts
12.      best_cfg ← i.placed
13.    continue
14.  // Branching: coba semua arah valid
15.  for dir in {UP, DOWN, LEFT, RIGHT}:
16.    next ← i.placed[-1] + dir
17.    // Pruning kriteria 2: self-avoiding
18.    if next ∈ i.occupied: continue
19.    // Hitung contact baru
20.    new_c ← countNewContacts(next, i, S)
21.    // Buat child state
22.    child.placed ← i.placed + [next]
23.    child.occupied ← i.occupied + {next}
24.    child.level ← i.level + 1
25.    child.contacts ← i.contacts + new_c
26.    // Hitung  $\hat{c}(\text{child})$ 
27.    ub ← UB_remaining(child)
28.    child.cost ←  $-(\text{child.contacts} + \text{ub})$ 
29.    // Pruning kriteria 1 pada child
30.    if child.contacts + ub ≤ best: continue
31.    // Masukkan ke priority queue
32.    Q.insert(child)
33. return best, best_cfg
```

Pada program ini, digunakan juga fungsi pembantu *countNewContacts* yang digunakan untuk menghitung jumlah kontak H-H baru yang terbentuk ketika asam amino pada level *k* ditempatkan di posisi *next*:

```

function countNewContacts(next, state, S):
    if S[state.level] ≠ H: return 0
    count ← 0
    for each neighbor of next (4 arah):
        for i in 0..state.level-1:
            if state.placed[i] == neighbor:
                if S[i] == H and |i - state.level| > 1:
                    count++
    return count

```

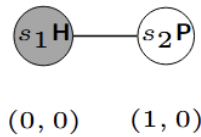
Algoritma diimplementasikan dalam bahasa C++ dengan memanfaatkan *Standard Template Library* (STL). Struktur data *priority_queue* dengan *comparator* berbasis nilai $\hat{c}(i)$ digunakan sebagai *live node queue*, dan *unordered set* digunakan untuk pengecekan ketersediaan sel pada kisi dalam waktu $O(1)$. Kode sumber lengkap dapat diakses pada tautan Github yang terdapat pada lampiran.

D. Contoh Penelusuran

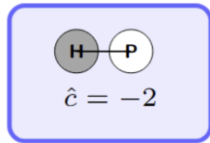
Untuk mengilustrasikan prosedur algoritma, digunakan sekuens pendek $S = \text{HPHH}$ ($n = 4$) dengan satu H-H *contact* optimal yang telah diverifikasi secara manual.

1. Inisialisasi

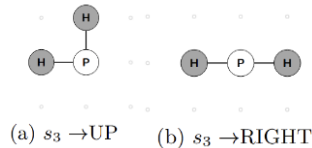
Mengikuti aturan eliminasi simetri pada Bagian II.E, dua asam amino pertama difiksasi posisinya. Pada langkah awal ini, *contacts* awal bernilai 0.



Berikut adalah *state space tree* yang terbentuk, yang hanya terdiri dari *root*:

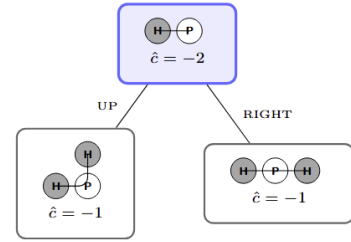


2. Dari posisi $s_2 = (1,0)$, eliminasi simetri membatasi arah yang dicoba hanya pada UP dan RIGHT (arah DOWN dan LEFT dieliminasi karena menghasilkan konformasi yang ekuivalen secara simetri).

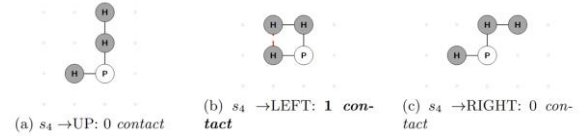


Kedua cabang menghasilkan nilai $\hat{c}(i)$ yang sama (-1), sehingga keduanya dimasukkan ke dalam *priority queue Q* dan tidak ada yang dipangkas pada tahap ini. Pemilihan cabang mana yang diekspansi lebih dahulu bersifat *arbitrary* di antara nilai $\hat{c}(i)$ yang

sama; pada penelusuran ini, cabang UP dipilih lebih dahulu.



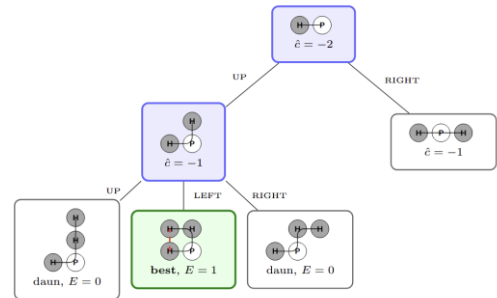
3. Dari posisi $s_3 = (1,1)$, terdapat tiga arah valid (arah DOWN tidak valid karena akan kembali ke posisi s_2 yang sudah terisi).



Ketiga cabang ini merupakan simpul daun ($\text{level} = n = 4$), sehingga langsung dibandingkan dengan *best_contacts_found*. Cabang LEFT menghasilkan kontak H-H baru antara $s_4 = (0,1)$ dan $s_1 = (0,0)$, dimana keduanya bertetangga pada kisi ($\| (0,1) - (0,0) \|_1 = 1$) dan tidak terhubung langsung pada rantai ($|4 - 1| = 3 > 1$), sehingga memenuhi batasan kedua pada Bagian II.C.

$$\Rightarrow \text{best_contacts_found} = 1$$

dengan konfigurasi $s_1 = (0,0)$, $s_2 = (1,0)$, $s_3 = (1,1)$, $s_4 = (0,1)$.



4. Setelah *best_contacts_found* ter-update menjadi 1, simpul hidup yang tersisa di Q , yaitu cabang $s_3 = \text{RIGHT}$ dari Tahap 2, dievaluasi kembali sebelum diekspansi:

$$\hat{c}(i) = -(\text{contacts_so_far} + \hat{g}(i)) = -(0 + 1) = -1$$

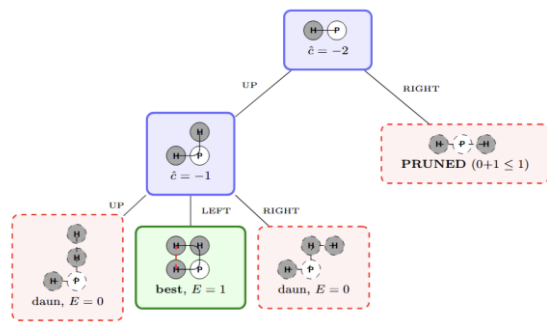
Kriteria *pruning* pertama diperiksa:

$$\text{contacts_so_far} + \hat{g}(i) = 0 + 1 = 1$$

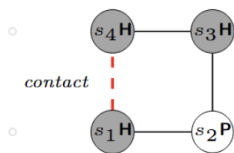
$$\leq \text{best_contacts_found} = 1$$

Karena kondisi pemangkasan terpenuhi (nilai batas atas tidak melampaui solusi terbaik yang sudah ditemukan), cabang $s_3 = \text{RIGHT}$ dipangkas tanpa perlu diekspansi lebih lanjut. Hal ini valid karena $\hat{g}(i) = 1$ merupakan *upper bound* sehingga jalur manapun yang melalui cabang ini dijamin tidak dapat

menghasilkan lebih dari 1 kontak H-H, sehingga tidak mungkin mengungguli solusi yang sudah ditemukan.



5. Dengan Q kosong setelah pemangkasan pada Tahap 4, algoritma berhenti dengan solusi optimal $E^* = 1$ dengan konformasi:



IV. HASIL DAN ANALISIS

A. Lingkungan Pengujian

Eksperimen dilakukan menggunakan implementasi BnB sesuai Algoritma 1 pada Bagian III, dikompilasi dengan GNU C++ Compiler versi 13.3.0 dengan tingkat optimasi -O2. Pengujian dijalankan pada satu *core* prosesor AMD Ryzen 5 5600H Laptop @ 3.3 Ghz. Sekuens uji terdiri dari enam sekuens HP dengan panjang $n = 8, 10, 12, 14, 16, 18$, yang dipilih untuk merepresentasikan variasi ukuran permasalahan sekaligus menjaga waktu eksekusi tetap pada rentang yang praktis untuk diamati.

B. Hasil Eksekusi

TABEL 1. HASIL EKSEKUSI BNB

n	Sekuens	E^*	Explored	Pruned	Waktu (s)
8	HPHPPHHP	2	98	85	0.000603
10	HPPHPPPHH	4	1.635	1.033	0.007793
12	HPHPPHHPH	5	18.358	17.271	0.11338
14	HHPHPPPHH	5	86.709	54.182	0.675295
16	PPPHHPPHH	5	993.584	604.116	7.873635
18	PHHHHPPHP	5	1.006.53	1.082.9	6.001848

Hasil pada Tabel I memperlihatkan pertumbuhan jumlah simpul yang dieksplorasi secara konsisten dengan kompleksitas

eksponensial $O(3^n)$ yang dianalisis pada Bagian II.E. Antara $n = 8$ dan $n = 16$, jumlah simpul yang dieksplorasi meningkat lebih dari 10.000 kali lipat, sejalan dengan pertambahan *branching factor* tiga pada setiap level tambahan kedalaman pohon.

C. Efektifitas Mekanisme Pruning

Pada eksperimen ini, juga dilakukan perbandingan dengan *brute force* tanpa pruning sama sekali, yakni 3^{n-2} .

TABEL 1. EFEKTIVITAS PRUNING TERHADAP BRUTE FORCE

n	Explored	3^{n-2}	Rasio Reduksi
8	98	729	7,4×
10	1.635	6.561	4,0×
12	18.358	59.049	3,2×
14	86.709	531.441	6,1×
16	993.584	4.782.969	4,8×
18	1.066.539	43.046.721	40,4×

Tabel II menunjukkan bahwa BnB secara konsisten mengeksplorasi jumlah simpul yang jauh lebih kecil dibandingkan estimasi *brute force* murni, dengan rasio reduksi berkisar antara 3,2 hingga 40,4 kali lipat pada rentang n yang diuji. Rasio reduksi pada $n = 18$ yang signifikan lebih tinggi (40,4×) mengindikasikan bahwa untuk sekuens dengan proporsi H rendah, *bound-based pruning* menjadi semakin efektif seiring bertambahnya n , sebuah sifat yang menguntungkan karena justru pada n besar pengurangan ruang pencarian paling dibutuhkan. Hal ini menguatkan posisi yang dibangun pada Bagian II.E, bahwa kombinasi *bound-based pruning*, *infeasibility pruning*, dan eliminasi simetri secara empiris menjaga BnB tetap praktis jauh di bawah batas teoritis $O(3^n)$, meskipun kompleksitas *worst-case* tetap eksponensial.

D. Analisis Pertumbuhan Waktu Eksekusi

Pertumbuhan waktu eksekusi terhadap n memperlihatkan tren eksponensial yang sejalan dengan kompleksitas *worst-case* $O(3^n)$. Meskipun demikian, laju pertumbuhan aktual secara konsisten lebih lambat dibandingkan batas teoritis tersebut, sebagai konsekuensi langsung dari efektifitas *pruning* yang dibahas pada Bagian IV.C. Karakteristik ini menjadi dasar argumen bahwa BnB merupakan pendekatan yang *layak* (*feasible*) untuk HP Model pada rentang n hingga sekitar 18–20 pada perangkat komputasi tunggal, sebagaimana juga dilaporkan oleh Hart dan Istrail [16] untuk implementasi BnB serupa pada HP Model.

V. KESIMPULAN DAN SARAN

Makalah ini telah memaparkan penerapan algoritma *Branch and Bound* (BnB) untuk menyelesaikan permasalahan *protein folding* pada *Hydrophobic-Polar* (HP) Model dua dimensi, dengan fungsi *cost* $\hat{c}(i) = \hat{f}(i) + \hat{g}(i)$. Fungsi *upper bound* $\hat{g}(i)$ berbasis pasangan H-H *eligible* yang tersisa terbukti valid sebagai batas atas yang aman, dan validasi terhadap penelusuran manual pada sekuens $S = \text{HPHH}$ menunjukkan

bahwa implementasi menghasilkan solusi optimal $E^* = 1$ yang konsisten dengan analisis teoritis.

Hasil eksperimen menunjukkan bahwa BnB secara konsisten mengeksplorasi jumlah simpul yang jauh lebih kecil dibandingkan estimasi *brute force* 3^{n-2} , dengan rasio reduksi berkisar 3,2 hingga 40,4 kali lipat pada rentang $n = 8$ hingga $n = 18$ yang membuktikan bahwa kombinasi *bound-based pruning*, *infeasibility pruning*, dan eliminasi simetri menjaga algoritma tetap praktis meskipun kompleksitas *worst-case* tetap eksponensial sebagai konsekuensi sifat NP-Complete permasalahan ini. Eksperimen juga mengungkap bahwa efektivitas *pruning* bergantung pada proporsi residu hidrofobik dalam sekuens, bukan semata panjangnya mengonfirmasi bahwa meskipun pendekatan berbasis *deep learning* seperti AlphaFold2 telah merevolusi prediksi struktur protein nyata, algoritma *exact* seperti BnB tetap relevan untuk konteks yang membutuhkan jaminan optimalitas matematis pada model yang dapat didiskretisasi seperti HP Model.

Penelitian lebih lanjut dapat diarahkan pada beberapa hal, yaitu berupa ekstensi algoritma ke kisi tiga dimensi agar lebih representatif terhadap fenomena *folding* biologis nyata, perancangan fungsi *upper bound* yang lebih ketat dengan memperhitungkan batasan *connectivity* pada sisa rantai agar *pruning* lebih agresif untuk sekuens dengan n besar, serta validasi terhadap sekuens HP yang diturunkan dari data protein nyata pada *Protein Data Bank* untuk mengamati korespondensi kualitatif antara solusi optimal BnB dan struktur tersier protein yang sesungguhnya.

PRANALA VIDEO YOUTUBE

<https://youtu.be/LQGSHEIVLxY>

PRANALA REPOSITORY GITHUB

https://github.com/YavieAzka/makalah_stima

REFERENSI

- [1] C. B. Anfinsen, "Spontaneous formation of the three-dimensional structure of proteins," *Dev. Biol.*, vol. 2, pp. 1–20, 1968. Chaudhuri TK, Paul S. Protein-misfolding diseases and chaperone-based therapeutic approaches. *FEBS J.* 2006 Apr;273(7):1331–49. doi: 10.1111/j.1742-4658.2006.05181.x. PMID: 16689923.
- [2] K. Chaudhuri and S. Paul, "Protein-misfolding diseases and chaperone-based therapeutic approaches," *FEBS J.*, vol. 273, no. 7, pp. 1331–1349, Apr. 2006.
- [3] R. Zwanzig, A. Szabo, and B. Bagchi, "Levinthal's paradox," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 89, no. 1, pp. 20–22, Jan. 1992.
- [4] B. Berger and T. Leighton, "Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete," *J. Comput. Biol.*, vol. 5, no. 1, pp. 27–40, 1998.

- [5] J. Jumper *et al.*, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021.
- [6] K. F. Lau and K. A. Dill, "A lattice statistical mechanics model of the conformational and sequence spaces of proteins," *Macromolecules*, vol. 22, no. 10, pp. 3986–3997, Oct. 1989.
- [7] R. Munir, "Algoritma branch and bound (bagian 1)," Slide Kuliah Strategi Algoritma, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung, Bandung, Indonesia, 2026. [Online]. Available: [https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2025-2026/17-Algoritma-Branch-and-Bound-\(2026\)-Bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2025-2026/17-Algoritma-Branch-and-Bound-(2026)-Bagian1.pdf)
- [8] A. L. Lehninger, D. L. Nelson, and M. M. Cox, *Lehninger Principles of Biochemistry*, 8th ed. New York, NY, USA: W. H. Freeman, 2021.
- [9] F. Chiti and C. M. Dobson, "Protein misfolding, functional amyloid, and human disease," *Annu. Rev. Biochem.*, vol. 75, pp. 333–366, 2006.
- [10] K. A. Dill and J. L. MacCallum, "The protein-folding problem, 50 years on," *Science*, vol. 338, no. 6110, pp. 1042–1046, 2012.
- [11] K. A. Dill, "Theory for the folding and stability of globular proteins," *Biochemistry*, vol. 24, no. 6, pp. 1501–1509, 1985.
- [12] R. Mavrevski, M. Traykov, N. Yanev, and B. Yurukov, "Protein folding in 3D lattice HP model using heuristic algorithm," *WSEAS Trans. Circuits Syst.*, vol. 18, 2018.
- [13] E. Sandelin, "On hydrophobicity and conformational specificity in proteins," *Biophys. J.*, vol. 86, no. 1, pp. 23–30, Jan. 2004.
- [14] R. Unger and J. Moult, "Genetic algorithms for protein folding simulations," *J. Mol. Biol.*, vol. 231, no. 1, pp. 75–81, May 1993.
- [15] E. L. Lawler and D. E. Wood, "Branch-and-bound methods: A survey," *Oper. Res.*, vol. 14, no. 4, pp. 699–719, 1966.
- [16] W. E. Hart and S. C. Istrail, "Fast protein folding in the hydrophobic-hydrophilic model within three-eighths of optimal," *J. Comput. Biol.*, vol. 3, no. 1, pp. 53–96, 1996.
- [17] P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, and M. Yannakakis, "On the complexity of protein folding," *J. Comput. Biol.*, vol. 5, no. 3, pp. 423–466, 1998.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 19 Juni 2026



Ttd
Yavie Azka Putra Araly
13524077