Università degli Studi di Padova

Facoltà di Ingegneria

Corso di Laurea Specialistica in Ingegneria Informatica

Un Algoritmo Genetico per la predizione della configurazione spaziale del nucleo idrofobico di proteine

Relatore : Prof. Ferrari Carlo

Laureando : Martino Fantato

Padova , Aprile 2010

Anno Accademico : 2009/2010

Indice generale

Introduzione

Il lavoro di questa tesi, si inserisce in un'area di interesse che interseca la bioinformatica e l'intelligenza artificiale. Nello specifico, si è affrontato il problema del *protein folding* (ripiegamento proteico) su un modello semplificato della proteina.

Lo studio del problema è iniziato esaminando le varie modellazioni del problema esistenti nella letteratura scientifica. Dopo lo studio teorico, è stato formulato un possibile metodo risolutivo a tale problema, mettendo in risalto le considerazioni che sono state fatte per la sua formulazione.

L' approccio risolutivo in questione è basato sull'algoritmo genetico nella sua definizione più classica, a cui viene integrato, nella fase di selezione della popolazione, il metodo *Probabilistic Roadmaps* (PRM, metodologia tipicamente usata nell'ambito della Robotica). Si sfrutta il metodo PRM, per potere “navigare” sulla superficie di energia libera associata alla popolazione (le conformazioni della proteina) dell'algoritmo genetico, in modo tale da poter guidare il processo verso l'ottimo con maggiore rapidità, rispetto alle prestazioni dell'algoritmo genetico classico. In altre parole, l'algoritmo proposto è una algoritmo genetico modificato, ovvero un algoritmo genetico *ibrido*.

La modellazione della proteina è fatta su un modello minimalista, detto modello HP. Tale modello studia solamente le interazioni idrofobiche che avvengono nel processo di folding tra gli amminoacidi che compongono la proteina. Questo non permette, quindi, una predizione completa della struttura finale della proteina a seguito del processo di folding, ma consente di individuare il nucleo idrofobico della proteina.

Il lavoro di tesi, giunge quindi ad individuare una possibile soluzione della struttura del nucleo idrofobico che una proteina presenta al termine del processo di folding , grazie allo studio della struttura della proteina stessa ed inoltre allo studio dell'energy landscape ad essa associata.

La prima parte della tesi parte è un' introduzione del problema e della sua modellazione (aspetto biologico, aspetto fisico e aspetto informatico), si passa poi ad analizzare, separatamente, gli algoritmi utilizzati nella risoluzione. Descritto questo, la parte teorica della tesi lascia spazio alla descrizione della parte implementativa del lavoro di tesi.

L'implementazione è stata fatta in codice C++ con l'utilizzo di due librerie GAlib e

BIU. La prima libreria è relativa all'uso di algoritmi genetici ed è stata implementata nel Massachusetts Institute of Technology; mentre la seconda libreria riguarda la modellazione della proteina su reticolo ed è stata implementata nell'Università di Friburgo.

Infine, vengono esposti i risultati della simulazione e le relative osservazioni, con le conclusioni finali.

Capitolo 1

Le Proteine : Struttura e il Protein Folding Problem

1.1 Dal genoma al protein folding

Il sequenziamento del patrimonio genetico di alcuni organismi, principalmente quello umano, assieme allo sviluppo e ai progressi nei metodi e nelle tecnologie di analisi, hanno aperto nuovi scenari conferendo alle proteine un ruolo sempre più importante e suscitando un interesse sempre maggiore presso la comunità scientifica internazionale, rendendo necessaria la nascita di un'ontologia che permettesse di riferirsi al nuovo campo di ricerca: la *proteomica*.

La proteomica è una disciplina scientifica che studia il *proteoma*[1], essa mira ad identificare le proteine ed ad associarle con uno stato fisiologico in base all'alterazione del livello di espressione fra controllo e trattato. Permette di correlare il livello di proteine prodotte da una cellula o tessuto e l'inizio o la progressione di uno stato di stress.

La proteomica assieme alla genomica, ricopre un ruolo fondamentale nella ricerca biomedica e, in futuro, avrà un impatto significativo sullo sviluppo dei sistemi diagnostici debellando patologie quali il morbo di Alzheimer e le neoplasie.

All'interno di questa area di ricerca, il problema del *protein folding* (ripiegamento proteico) è molto importante ed interessante. Questo lavoro di tesi, si concentrerà sullo studio di una metodologia per una possibile soluzione (parziale) al problema.

1 Il termine **proteoma**, coniato da Mark Wilkins nel 1995, è usato per descrivere l'insieme delle proteine di un organismo o di un sistema biologico, ovvero le proteine prodotte dal genoma.

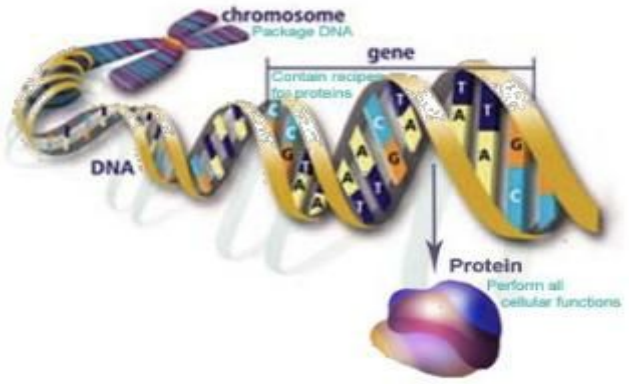


Figura 1.1 Dal cromosoma alla proteina

La centralità del problema del ripiegamento proteico in ambito medico, chimico e bioinformatico è data dal fatto che ottenere la struttura proteica per via sperimentalmente, ovvero tramite tecniche di laboratorio, è molto costoso e lento. Le tecniche per determinare la struttura della proteina in laboratorio sono due, la diffrazione dei raggi X e la spettroscopia a risonanza magnetica nucleare.

Tuttavia per la predizione della struttura proteica *in silicio* esistono numerosi fattori che la rendono molto difficile e complicata. Le principali problematiche sono relative all'elevato numero di strutture proteiche possibili; e che le conoscenze fisico-chimiche fin'ora conosciute riguardo la stabilità strutturale della proteina non sono del tutto definite chiaramente. Di conseguenza, qualsiasi metodo previsione delle struttura delle proteine necessita di una metodologia per esplorare lo spazio delle possibili strutture in maniera efficiente (una strategia di ricerca), e una metodologia per individuare la struttura più plausibile (una funzione energetica).

1.2 Introduzione sulle proteine

Le proteine sono eteropolimeri lineari, costituite da sequenze non periodiche di amminoacidi connessi da legami covalenti. La lunghezza di tali sequenze varia da circa 40 a più di 1000 amminoacidi. Gli amminoacidi sono costituiti da un gruppo peptidico e da un residuo (*side chain*). I legami covalenti tra i gruppi peptidici formano il *backbone* della proteina. I gradi di libertà rotazionali dei legami peptidici rendono il backbone molto flessibile e per questo fatto una proteina può essere pensata come un polimero (o più propriamente, data la diversità tra le unità, un eteropolimero).

Il lavoro indispensabile per attivare la funzione fisiologica di una proteina è svolto dal processo di ripiegamento (folding), durante il quale la proteina in soluzione si assesta su una struttura tridimensionale (detto anche stato terziario) che dev'essere autoconsistente con il solvente [1].

La struttura terziaria è stabilizzata da diverse interazioni, covalenti e non: ponti disolfurici, legami idrogeno, interazioni elettrostatiche e interazioni idrofobiche; esse si realizzano anche, e soprattutto, tra atomi lontani tra loro nella sequenza lineare e la struttura proteica è il risultato dell'equilibrio che si instaura tra queste forze. Le interazioni idrofobiche sono le principali responsabili dell'avvio del processo di ripiegamento [2] : una catena polipeptidica posta in un mezzo acquoso tende infatti ad avvolgersi in modo da formare un nucleo interno di gruppi idrofobici "nascosti" al solvente.

L'acquisizione della configurazione spaziale solubile, che le permetta di espletare le sue funzioni, è il "problema" che una proteina deve risolvere. Il raggiungimento della forma viene in parte resa possibile e ad ogni modo aiutata da proteine chiamate *chaperonine.* Il meccanismo del ripiegamento non è ancora del tutto chiaro, tuttavia è noto come l'aiuto da parte di chaperonine sia essenziale per tale processo.

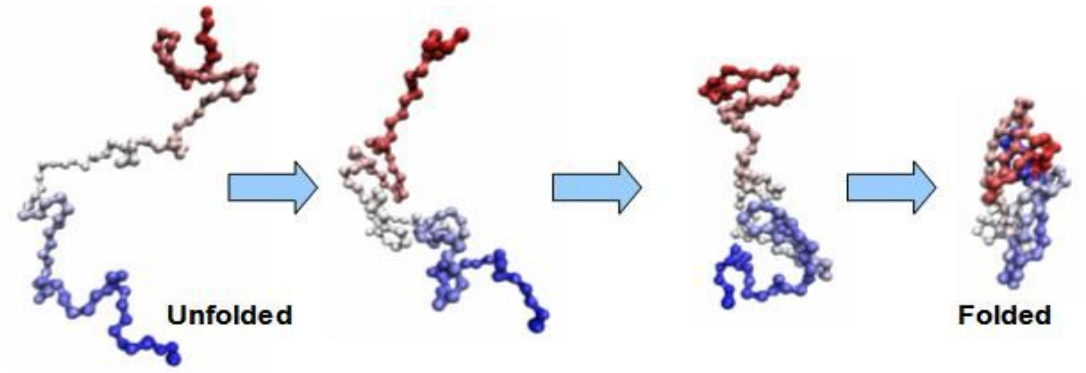


Figura 1.2 Processo di ripiegamento della proteina

1.3 Gli amminoacidi

Nelle proteine in natura troviamo 20 diversi tipi di amminoacidi, che sono i "mattoni" che le formano, e le diversità funzionali che le migliaia di proteina hanno sono dovute alle proprietà intrinseche degli amminoacidi.

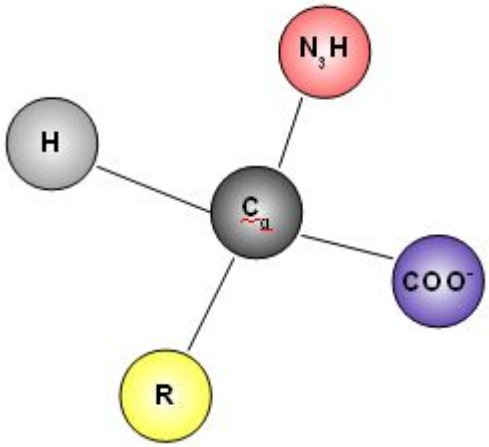
Le peculiarità chimiche che gli amminoacidi sono la capacità di polimerizzare[2], le proprietà acido-base, la variabilità di struttura e di funzionalità chimica nelle catena laterali amminoacidiche e la chiralità[3] (ad esclusione della glicina).

La struttura di un amminoacido isolato è composta da una parte centrale di carbonio alfa (C_α) tetraedrico, legato in modo covalente sia al gruppo amminico (NH_3^+) che al gruppo carbossilico (COO^-). Inoltre, legato alla parte centrale C_α, ci sono anche un atomo di idrogeno e un residuo laterale variabile, tale residuo conferisce la propria identità all'amminoacido.

2 Con il termine **polimerizzazione** si intende la reazione chimica che porta alla formazione di una catena polimerica, ovvero di una molecola costituita da molte parti uguali (detti "monomeri" o "unità ripetitive") che si ripetono in sequenza.

3 In chimica, una molecola che ammette un'immagine speculare non sovrapponibile a sé è detta **chirale**. Al contrario, una molecola che invece è sovrapponibile alla propria immagine speculare è detta **achirale**.

1.3Gli amminoacidi



C_α carbonio alfa

NH_3^+ gruppo amminico

COO^- gruppo carbossilico

R residuo laterale variabile

H atomo di idrogeno

Figura 1.3 Struttura Amminoacido tetraedrica

Gli amminoacidi si possono unire tra loro attraverso legami peptidici, ed è quanto avviene nelle proteine. La presenza di due gruppi chimici caratteristici (NH_3^+ e COO^-), permette agli amminoacidi di polimerizzare e quindi, formare peptidi e proteine. Il gruppo amminico e il gruppo carbossilico possono reagire in maniera testa-coda, perdendo una molecola d'acqua e formando un legame amminico covalente; questo nel caso di peptidi e proteine, viene chiamato legame peptidico. Il ripetersi di questa reazione produce polipeptidi e proteine.

Nella figura 1.2 sono riportate le strutture di tutti gli amminoacidi, con la seguente convenzione: in grigio sono visualizzati gli atomi in carbonio, in blu quelli di azoto, in bianco quelli di idrogeno e in rosso le molecole di ossigeno.

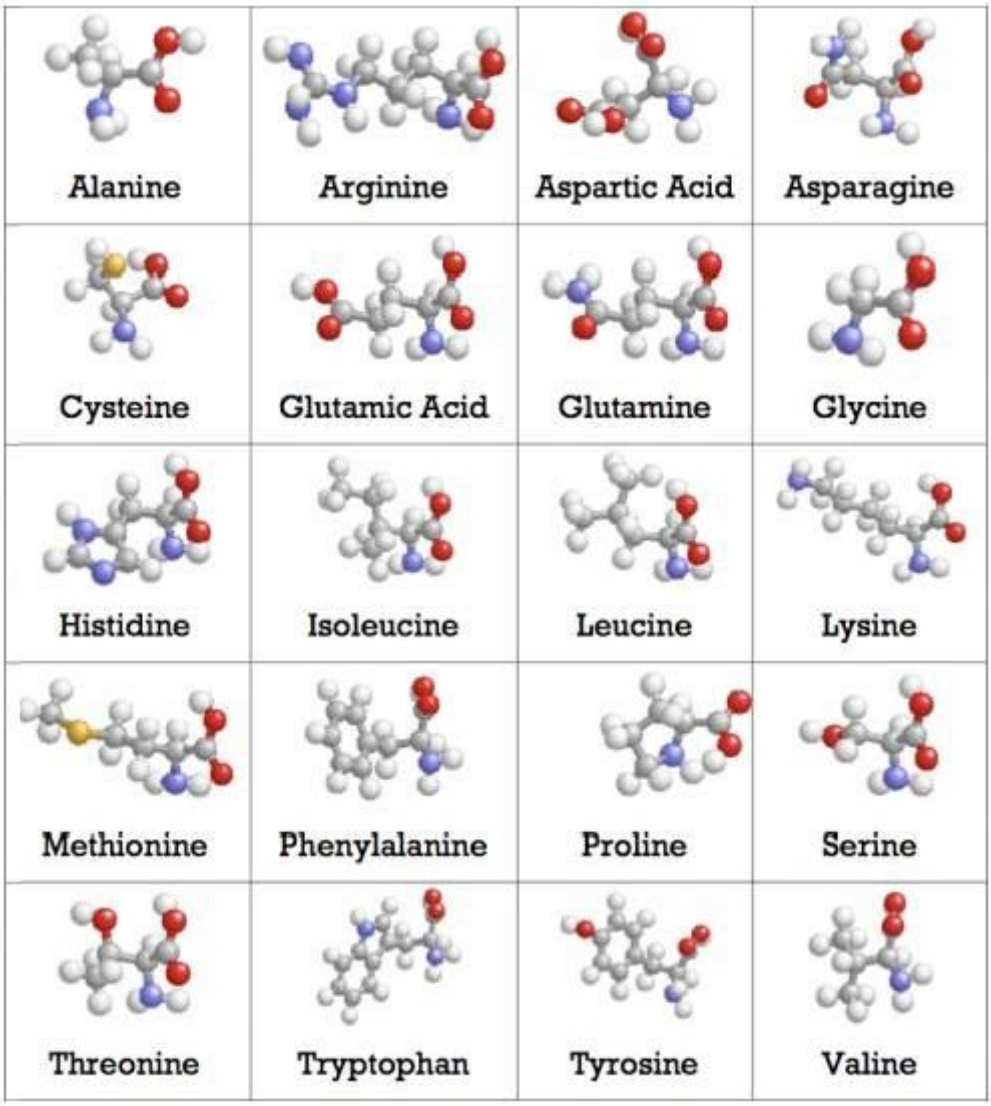


Figura 1.4 Struttura chimica degli amminoacidi

1.4 Le proprietà strutturali della proteina

Le proteine dal punto di vista chimico sono : polimeri non ramificati di amminoacidi legati in maniera testa-coda, dal gruppo carbossilico al gruppo amminico, attraverso la formazione di un legame peptidico covalente, con perdita di una molecola di acqua.

Lo scheletro strutturale della proteina è rappresentato dalla sequenza ripetuta -N-C_α -CO- dove N rappresenta l'azoto ammidico, C_α è il carbonio-α di un amminoacido nella catena polimerica, e il C finale è il carbonio carbonilico dell'amminoacido, che a sua volta è legato a N ammidico dell'amminoacido seguente lungo la sequenza.

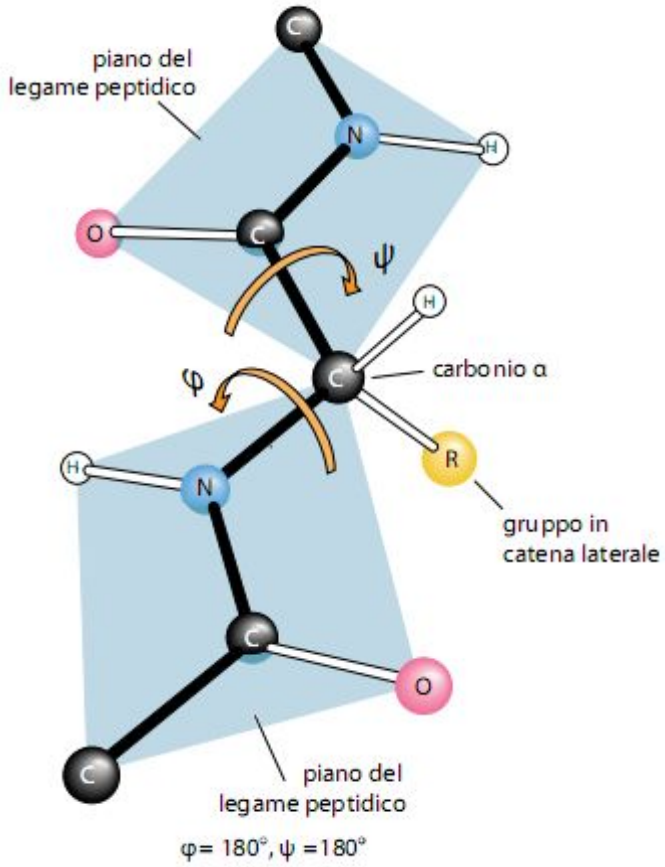


Figura 1.5 Struttura della proteina

Il legame peptidico è solitamente rappresentato da un legame singolo tra il carbonio carbonilico e l'azoto ammidico. Perciò si possono avere rotazioni intorno a ciascun legame

covalente dello scheletro peptidico. In tal caso, l'azoto ha un coppia di elettroni non appaiati in un orbitale.

Tuttavia è possibile un'altra forma di risonanza per il legame peptidico, in cui C e N lasciano una coppia non appaiata di elettroni sull'ossigeno. Tale struttura, avendo un legame doppio, impedisce le rotazioni attorno al legame peptidico. Nella realtà il legame peptidico è interne tra questi esterni; e possiede un carattere parziale di legame doppio.

1.4 Livelli di struttura

La proteina può essere paragonata ad una struttura tridimensionale articolata su 4 livelli, in relazione fra di loro.

1. *Struttura Primaria*: Corrisponde alla specifica sequenza degli amminoacidi del backbone.
2. *Struttura Secondaria*: I diversi amminoacidi spesso formano delle strutture geometriche ordinate localizzate in parti della proteina. Tali strutture, fra le quali riconosciamo le α-eliche e le β-sheets, sono dette motivi di struttura secondaria.
3. *Struttura Terziaria*: Costituisce la vera struttura tridimensionale della proteina. Essa corrisponde alla struttura assunta dalla proteina quando essa si trova nel cosiddetto stato nativo.
4. *Struttura Quaternaria*: Riguarda in genere proteine molto grandi. In effetti, spesso tali proteine sono costituite da varie sub-unità essenzialmente uguali fra loro. La struttura quaternaria riguarda la disposizione spaziale e topologica di queste sub-unità.

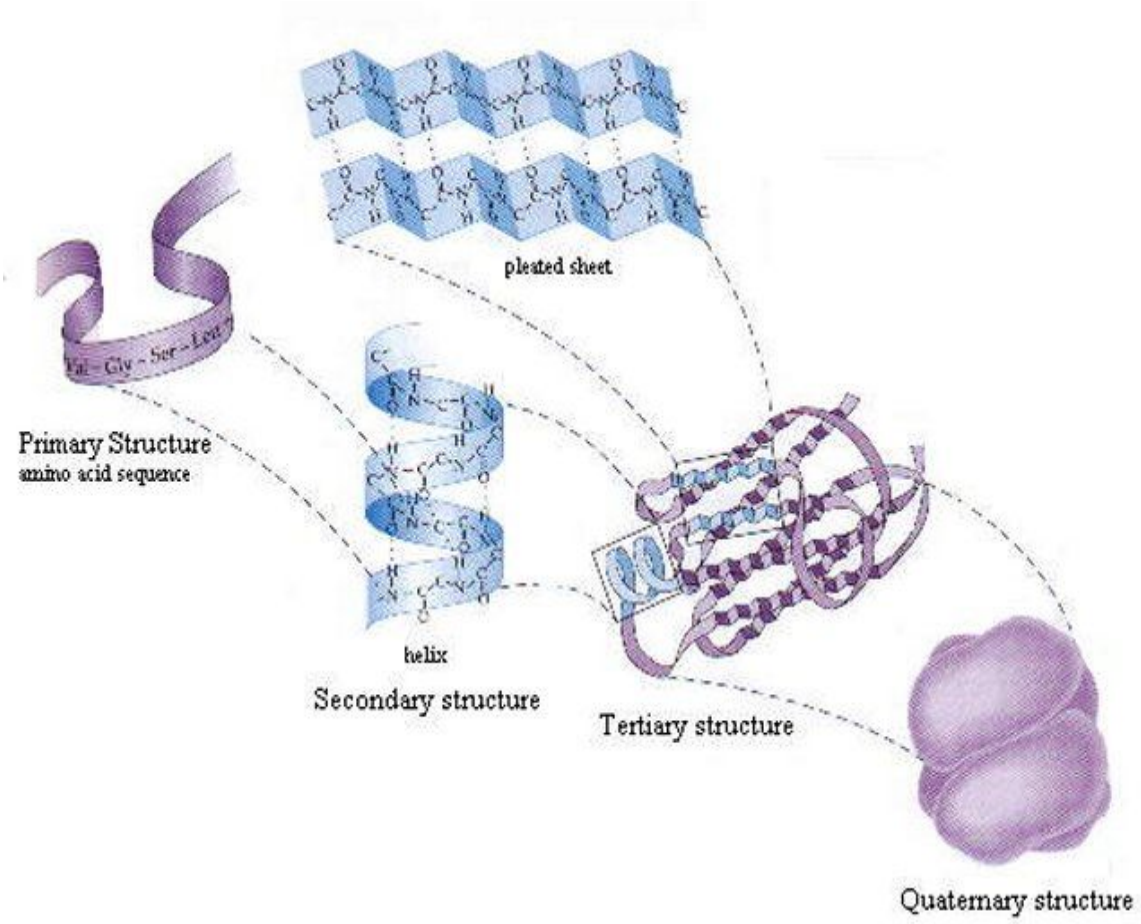


Figura 1.6 Livelli di struttura della proteina

1.5 Il problema del Protein Folding

Il problema del ripiegamento proteico usando metodi ab initio può essere descritto come segue: predire la configurazione tridimensionale (struttura terziaria) che assume una proteina partendo solamente dalla sequenza amminoacidica (struttura primaria).

Nella figura 1.7 si mostra una schematizzazione grafica del problema [1].

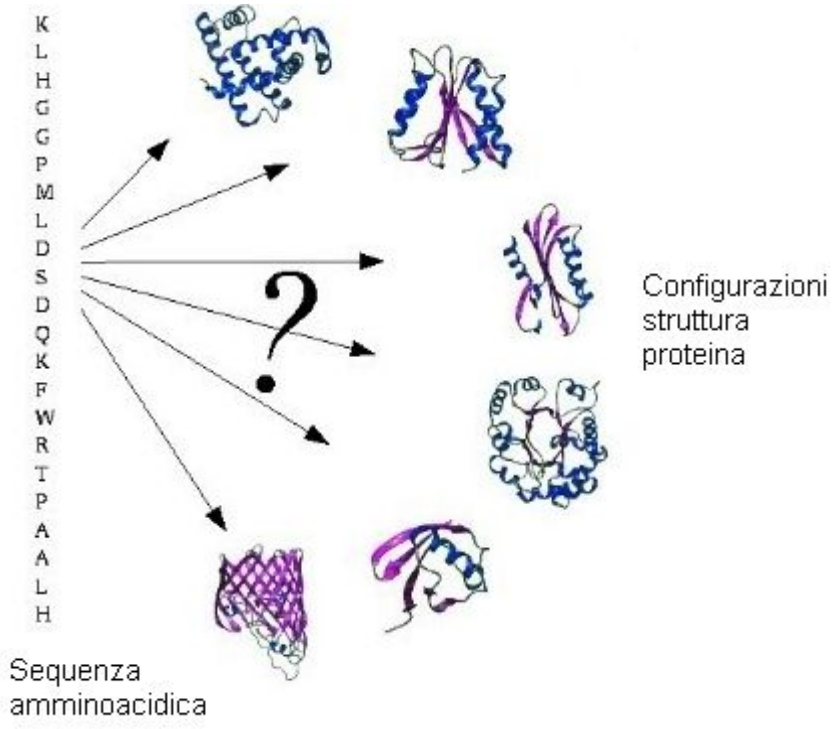


Figura 1.7 Protein Folding Problem

Una delle ipotesi fondamentali su cui si basa la teoria del protein folding è il dogma di Anfinsen [3]. Questo postulato della biologia molecolare fu elaborato da Christian B. Anfinsen e collaboratori alla fine degli anni 1950 nel dipartimento National Institutes of Health di Harvard. Essi dimostrarono che molte proteine si possono spontaneamente ripiegare (in vitro) dopo essere state completamente denaturate, concludendo che la struttura tridimensionale deve essere completamente determinata dalla struttura primaria: non era necessario chiamare in causa alcun fattore esterno.

Il dogma di Anfinsen, conosciuto anche come ipotesi termodinamica di Anfinsen, valse il premio Nobel per la Chimica al professore che le ha dato il nome, Christian B. Anfinsen appunto. Il dogma afferma che, almeno per le piccole proteine globulari, la struttura nativa è determinata solamente dalla sequenza di amminoacidi che costituiscono la proteina. Ciò equivale a dire che, nelle condizioni ambientali (temperatura, concentrazione del solvente e composizione, ecc.) alle quali avviene il folding proteico, la struttura nativa corrisponde a un unico minimo di energia libera, stabile e cineticamente accessibile.

Analizzando le tre condizioni relative al minimo di energia libera:

- *Unicità*: richiede che la sequenza non possieda qualche altra configurazione

dotata di energia libera comparabile. Quindi il minimo di energia libera deve essere univoco.

- *Stabilità*: piccoli cambiamenti nell'ambiente circostante non possono produrre cambiamenti nella configurazione a energia minima. Ciò può essere descritto come una superficie parabolica di energia libera con lo stato nativo corrispondente al punto di minimo; la superficie di energia libera nelle vicinanze dello stato nativo deve essere piuttosto ripida ed elevata, per potere fornire la stabilità.
- *Accessibilità cinetica*: significa che il percorso nella superficie di energia libera dallo stato denaturato a quello con riavvolgimento proteico deve essere ragionevolmente piano o in altre parole che il ripiegamento della catena non deve implicare cambiamenti altamente complessi nella forma (come nodi o altre conformazioni di ordine superiore).

Come la proteina raggiunga questa struttura rientra nel campo di studio del folding proteico, che si basa un altro dogma correlato chiamato paradosso di Levinthal.

Cyrus Levinthal, grazie ai suoi studi [4], nel 1968 notò che a causa dell'elevato numero di gradi di libertà di un polipeptide non ripiegato (la proteina denaturata), questa avrebbe un numero incredibilmente grande di possibili configurazioni strutturali finali, secondo quanto stimato dallo stesso Levinthal, circa 10^{300}. Se la proteina raggiungesse la sua configurazione finale passando via via attraverso tutte le possibili configurazioni, anche solo per brevi istanti, sarebbe necessario un tempo ben superiore all'età attualmente stimata dell'universo per raggiungere la configurazione corretta. In realtà, molte piccole proteine si ripiegano spontaneamente in un tempo dell'ordine dei millisecondi o addirittura dei microsecondi.

La differenza enorme che esiste tra il tempo del foldig prevedibile in teoria e quello osservato in realtà è appunto chiamato paradosso di Levinthal. Il paradosso di Levinthal, in effetti, è stato spesso frainteso come una teoria malriuscita sul folding proteico. Una delle teorie che sono emerse per superare questo paradosso è quella che prevede che tutte le possibili configurazioni energetiche formino un profilo ad imbuto (*funnel landscape*), che favorisce la scelta della conformazione corretta in tempi rapidi [5].

Basandoci sul paradosso di Levinthal, questo rende la predizione computazionale della struttura proteica tramite la valutazione di tutte le possibili configurazioni irrealizzabile persino per proteine relativamente piccole. Inoltre, dalla teoria della Biologia si sa che alcune proteine necessitano dell'assistenza di un'altra proteina chiamata chaperonina per realizzare un

avvolgimento corretto. Questa osservazione andrebbe a contraddire il dogma di Anfinsen, tuttavia le chaperonine non sembrano influenzare lo stato finale della proteina, ma dovrebbero agire principalmente prevenendo l'aggregazione di diverse molecole proteiche prima che la proteina vada incontro al ripiegamento. Da ciò Anfinsen dedusse anche che la stabilità termodinamica della struttura ripiegata correttamente rappresenta la guida del processo di folding, e che non vi è alcuna influenza su questa da parte delle chaperonine. La teoria del protein folding nonostante la presenza delle chaperonine e il paradosso di Levinthal, risulta comunque valida. Per questo motivo in questa tesi si considera vera la seguente affermazione: il termine del processo di protein folding è il raggiungimento dello stato nativo, dove per stato nativo si intende la conformazione più stabile di una proteina [6].

1.6 Gli approcci risolutivi del Protein Folding Problem

Una proteina può essere schematizzata come una sequenza di elementi, detti amminoacidi, che sotto determinate condizioni fisiche, si ripiegano (*folded*) in una unica struttura funzionale chiamata stato nativo, o struttura terziaria. Trovare lo stato nativo partendo solo dalla sequenza lineare di amminoacidi, così viene definito il **Protein Folding Problem**.

Per provare a creare una metodo di soluzione al problema, occorre valutare con attenzione le varie metodologia e tecniche di approccio al problema in questione. La scelta tra queste deve essere effettuata in base allo scopo finale che si vuol raggiungere. Verranno presentate di seguito, le varie tecniche e metodologie per affrontare il problema.

Esistono tre metodi principali di predizione della struttura terziaria [1]:

1. *Homology Model* : si basa su una considerazione intuitiva e ragionevole, ovvero che due proteine con sequenza amminoacidica similare condivideranno strutture anch'esse molto simili. La similitudine tra le proteine sono individuate attraverso l'allineamento delle sequenze. Tuttavia il principale collo di bottiglia nella modellazione comparativa nasce dalla difficoltà di allineamento, e dalla complessità dei tale calcolo. Ovviamente il risultato sarà tanto più preciso quanto più le due sequenze simili.
2. *Fold Recognition (Threading)* : questo metodo scansiona la sequenza di amminoacidi di struttura sconosciuta con tutte le strutture note presenti in un database. Ad ogni scansione viene assegnato un punteggio per valutare la compatibilità della sequenza di amminoacidi alla struttura nota, ottenendo così un insieme di possibili modelli

tridimensionali. Questo tipo di metodo è conosciuto anche come il metodo dei profili 1D-3D, che deriva dal verificare la compatibilità tra sequenze proteiche lineare e strutture tridimensionali.

3. *Ab initio* : questi metodi di modellazione cercano di costruire modelli tridimensionali di proteine partendo "da zero", cioè le conoscenze su cui basare la predizione della struttura sono solamente la sequenza amminoacidica e le conoscenze teoriche chimico-fisiche, piuttosto che su strutture proteiche già note. Normalmente queste procedure tendono a richiedere grandi risorse di calcolo, e sono quindi utilizzati solo per le piccole proteine. Per la predizione di proteine più grandi si richiedono algoritmi ottimizzati per il calcolo parallelo o distribuito. Sebbene le risorse di calcolo necessarie siano grandi, i metodi ab initio, sono una parte molto attiva e importante all'interno della ricerca sul protein folding.

I metodi visti, potrebbero prestarsi ad un'ulteriore divisione, in sole due categorie, i metodi che sfruttano le strutture note e le conoscenze teoriche chimico-fisiche (Homology Model e Fold Recognition) e i metodi che sfruttano solamente queste ultime nozioni (Ab initio).

Dopo aver descritto su quali principi basare la predizione, si deve individuare quale approccio al problema adottare. Fino ad ora ci sono tre approcci al problema, i primi due di questi, son quelli più considerati dagli studi attuali [7].

1. L'*approccio termodinamico*. In questo approccio, la conformazione amminoacidica è studiata in termini di *energia libera*, l'ipotesi iniziale su cui si basa questo approccio è che lo stato nativo è il solo che minimizza l'energia libera.
2. L'approccio *dinamico*. In questo caso l'ipotesi base è l'esistenza di un *"folding tunnel"* che guida la proteina in un unico e stabile stato, lo stato nativo appunto. Sono usati gli stessi concetti della termodinamica, ma in maniera differente.
3. Un approccio alternativo ipotizza che il meccanismo di folding è *codificato* nella proteina mediante un linguaggio sconosciuto.

Infine, si può modellare la proteina in due maniere differenti : *all-atom* o *minimalista*. La scelta tra questi due differenti modelli è guidata dalle diverse finalità di predizione, si si vuole eseguire una predizione completa della proteina si dovrà usare il modello all-atom, mentre nel caso predizione di parti o componenti di proteina si può usare la modellazione minimalista [1].

1. Modello *All-Atoms* : le proteine sono considerate mediante tutti i loro atomi e le forze in azione. Questa è la simulazione di dinamica molecolare più comune effettuata dagli scienziati ma è anche la più costosa. Il vantaggio è che si tiene conto di tutte le caratteristiche delle molecole del sistema (nello specifico l'acqua intorno alla proteina è di solito molto importante) ma il costo computazionale è molto alto.
2. Modello *Minimalista* : la proteina viene semplificata, da risultare una schematizzazione molto astratta di una proteina reale. Tuttavia, anche se con queste caratteristiche, tali modelli sono tuttora molto utilizzati per lo studio del protein folding: ciò è dovuto sia al fatto che la notevole semplificazione porta alla fattibilità di calcoli molto pesanti e complessi per altra via, sia perché i modelli in questione racchiudono comunque aspetti e caratteristiche capaci di descrivere molte delle proprietà essenziali nel protein folding.

In questa tesi si è voluto affrontare il problema del ripiegamento proteico attraverso l'uso di metodo di predizione *ab initio,* con un approccio *dinamico* e basato su un modello *minimalista*, detto modello HP. Nel corso della tesi verranno presentate le scelte fatte, tuttavia occorre mettere in risalto un'osservazione : la scelta del modello HP, è dettata dal fatto che si intende predire la struttura tridimensionale del nucleo idrofobico della proteina e non la struttura proteica tridimensionale effettiva. Le considerazioni fatte congiunte con la documentazione trovata nella letteratura scientifica riguardante il protein folding, auspicano che la tipologia di approccio scelta, sia, a priori, la più promettente per riuscire a trovare una soluzione con buoni risultati.

Capitolo 2

Il modello HP

Il modello HP, rientra nella categoria dei modelli minimalisti per la rappresentazione della struttura della proteina. Questi modelli sono semplici per rappresentare della struttura e possono essere definita da un reticolo. Nella rappresentazione sul reticolo (a 2 o 3 dimensioni), che è la rappresentazione scelta per questo lavoro, ogni posizione è riempita da al massimo un amminoacido; la corrispondenza tra amminoacidi e posizioni è chiamata *embedding* della proteina, e quando l'embedding è iniettivo, la conformazione che la proteina assume è chiamata *self avoiding*. Il problema, sotto queste ipotesi, è stato dimostrato essere NP-completo negli anni novanta da Patterson e Prytycka [8].

Perciò, l'esistenza di un algoritmo che risolve il problema esattamente senza questo modello è certo essere impossibile. L'uso di algoritmi euristici e approssimati diventa il più promettente percorso di risoluzione, o perlomeno, per alcune delle istanze del problema. Partendo da questi questioni teoriche e pratiche del problema, risulta utile spendere risorse e tempo nella modellazione del processo di folding.

Punto centrale del modello HP, e di questo stesso lavoro, è l'osservazione che le forze che agiscono maggiormente nel processo di folding sono le forze idrofobiche e la maggior parte delle proteine ripiegate hanno un nucleo idrofobico interno mentre le catene laterali sono polari, e si dispongono sulla superficie esposta con le molecole d'acqua circostanti, interagendo con esse. Inoltre, numerose teorie affermano che il processo di ripiegamento si avvicina allo stato nativo, riducendo al minimo il numero di amminoacidi idrofobici esposti alle molecole d'acqua circostanti [2, 9].

Il modello idrofobico-idrofilico (o polare), detto modello HP, è il modello che permette di formalizzare lo studio del protein folding in queste condizioni. Tale modello considera solo due tipi di amminoacidi : gli idrofobici, rappresentati come H, e gli idrofilici, rappresentati come P (polari).

Sebbene la divisione degli amminoacidi tra idrofobici e polari non sia univoca, in figura 2.1, viene proposta una divisione dei venti amminoacidi in polari e idrofobici.

Amminoacido			Side Chain		Amminoacido			Side Chain
Aspartic acid	Asp	D	negative		Aline	Ala	A	nonpolar
Glutamic acid	Glu	E	negative		Glycine	Gly	G	nonpolar
Arginine	Arg	R	positive		Valine	Val	V	nonpolar
Lysine	Lys	K	positive		Leucine	Leu	L	nonpolar
Histidine	His	H	positive		Isoleucine	Ile	I	nonpolar
Asparagine	As	N	uncharged polar		Proline	Pro	P	nonpolar
Glutamine	Gln	Q	uncharged polar		Phenylaline	Phe	F	nonpolar
Serine	Ser	S	uncharged polar		Methionine	Met	M	nonpolar
Threonine	Thr	T	uncharged polar		Tryptophan	Trp	W	nonpolar
Tyrosine	Tyr	Y	uncharged polar		Cysteine	Cys	C	nonpolar
Amminoacidi POLARI					**Amminoacidi IDROFOBICI**			

Figura 2.1 Amminoacidi Polari e Idrofobici

2.1 Definizione

Il modello HP è un modello NP-completo ad energia libera che è motivato dal ruolo centrale e importante degli amminoacidi idrofobici e idrofilici per la struttura proteica, introdotto da K.A.Dill nel 1985 [9]. Riassumendo brevemente, esso di basa sulle seguenti considerazioni :

1. L' interazione idrofobica è la forza guida per il folding proteico e l'idrofobicità degli amminoacidi è la forza maggiore per lo sviluppo della conformazione nativa per proteine globulari piccole.
2. La struttura nativa di molte proteine sono compatte e hanno un nucleo ben definito formato dai residui idrofobici che sono minimamente esposti al solvente, protetti dalle superficie di residui polari.

Nel modello standard di Dill, ogni amminoacido è rappresentato da un pallino ed i legami della catena amminoacidica sono rappresentati da linee. La proteina risulta essere composta da una specifica sequenza di solo due tipi di pallini, H (pallini-idrofobici/non polari) o P (pallini-idrofilici/polari). Si riduce l'alfabeto, con cui una stringa che identifica la sequenza amminoacidica può essere formata, da 20 caratteri a soli due. La sequenza della

proteina nel modello HP è quindi una stringa appartenente all'alfabeto $\{H, P\}^+$. Gli amminoacidi idrofobici tendono a avere una forma compatta al centro della conformazione (nucleo) che esclude la soluzione acquosa in cui è circondata la proteina. L' idrofobicità è uno dei fattori chiave che determina come la catena di amminoacidi si ripiegherà in una proteina attiva.

Quando il modello HP è su reticolo, tutta la conformazione è incorporata in una reticolo quadrato (2D) o cubico (3D), che divide semplicemente lo spazio in locazioni per gli amminoacidi. Gli angoli dei legami hanno un valore limitato e discreto, dettato dalla struttura del reticolo (per esempio, il reticolo quadrato e cubico hanno angoli di 90°, mentre il triagolare ha angoli di 60°, ecc..). Un posto del reticolo può essere vuoto o contenere un pallino (amminoacido). In particolare, nel reticolo cubico 3D, il modello HP rappresenta le proteine come una linea (la catena di amminoacidi) che non si interseca mai sé stesso, per esempio, due pallini non possono occupare la stessa posizione nel reticolo, e ogni pallino occupa solo una posizione nel reticolo connesso ai suoi vicini della catena di amminoacidi.

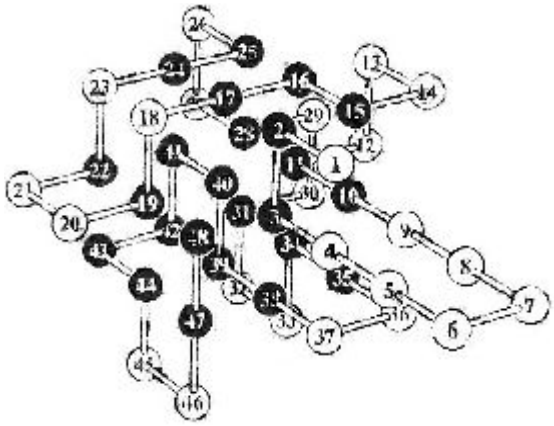


Figura 2.2 Esempio di modello HP su reticolo cubico

2.2 PFP nel modello HP

Il problema del folding proteico HP può essere formalmente definito come segue; data una sequenza di amminoacidi $s = s_1, s_2,..., s_n$, trovare una conformazione che minimizzi l'energia (ipotesi di Anfiensen), cioè, trovare $c^* \epsilon C(s)$ tale che $E^*=E(c^*)=min\{E(c)| c \epsilon C\}$,

dove *C(s)* è un insieme di tutte le conformazioni valide per *s* [10]. Il processo di folding della proteina su reticolo può essere riassunto dal disegno mostrato in figura 2.2.

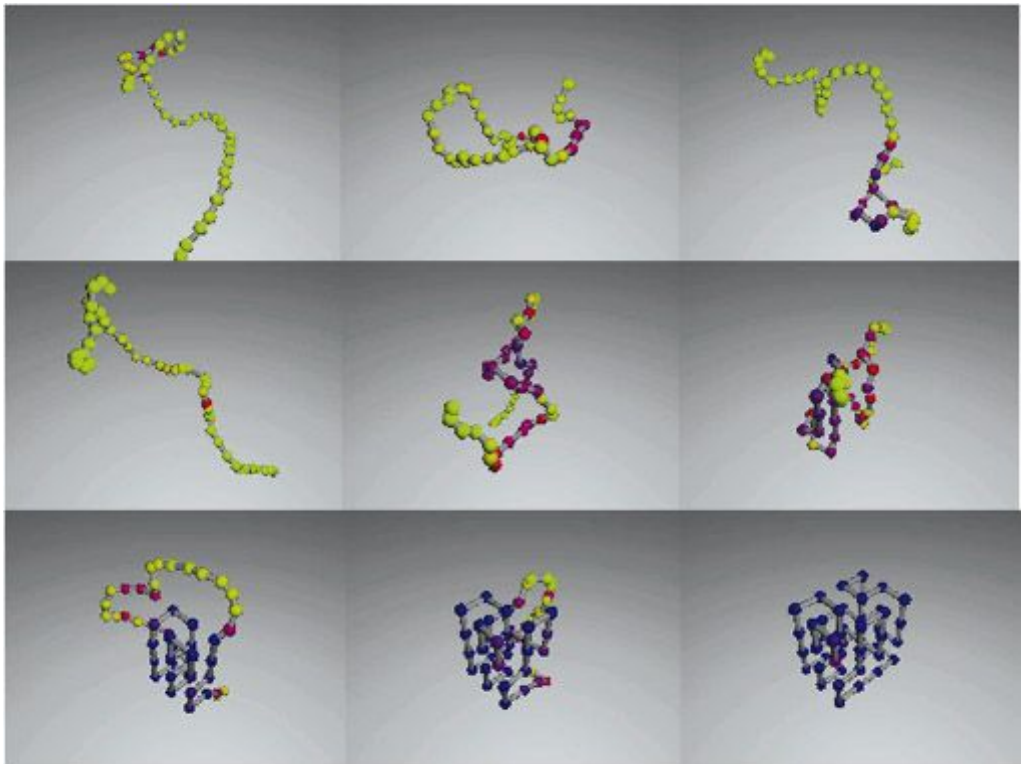


Figura 2.2 Protein Folding su reticolo

La funzione energia prende in considerazione solo le interazioni tra vicini topologici di tipo H, queste interazioni sono chiamate *legami*.

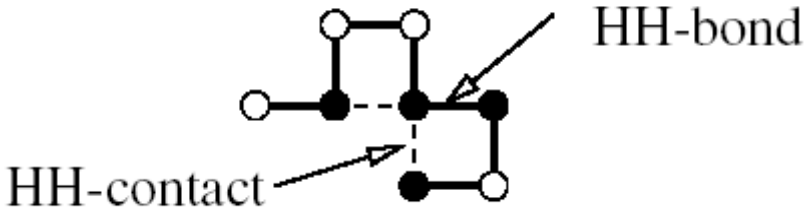


Figura 2.3 Contatto idrofobico e legame idrofobico

Nel caso 3D, il numero massimo di vicini è 6 e il massimo numero di vicini topologici è 4 e 5, rispettivamente per un residuo intermedio e uno terminale della sequenza.

Per ogni conformazione, si può calcolare il valore della funzione energia : questo permette la modellazione dell'energia libera del folding proteico. La più semplice forma di funzione di energia conta solo il numero di contatti idrofobici, ossia contatti H-H. Ogni

contatto topologico H-H ha un valore energetico ε, mentre tutti gli altri tipi di contatti (H-P , P-H, P-P) hanno valore energetico δ. Due amminoacidi creano un contatto H-H se sono topologicamente vicini e non sono connessi da un legame. Lo scopo è trovare una conformazione con il più basso livello energetico. Nel modello generale HP, le interazioni tra i residui possono essere definite come segue : $e_{HH} = -|\varepsilon|$, ed

$e_{HP} = e_{PH} = e_{PP} = \delta$.

Quando ε assume il valore -1 e δ il valore 0, si ha la tipica matrice di interazione energetica per il modello standard HP. La sua definizione è mostrata in figura 2.3 [11].

$$HP = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

Figura 2.4 Matrice HP

Per il modello di Dill, la conformazione nativa è quella che massimizza il numero di contatti H-H, o in modo equivalente, quella che minimizza la funzione di energia libera.

Il modello HP, individua e considera solamente i contatti H-H della struttura proteica. Questo si traduce che la predizione della struttura nel processo di folding, non sia relativa alla struttura della proteina completa, ma solamente riguarda il nucleo idrofobico della proteina stessa. Tuttavia, individuato il nucleo idrofobico della proteina, attraverso ulteriori elaborazioni si può ricostruire il backbone proteico completo. Ecco perché risulta interessante individuare la struttura idrofobica della proteina, nonostante sia una visione parziale della struttura complessiva [11, 12].

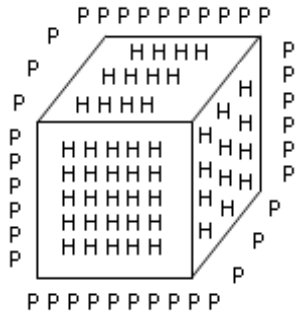


Figura 2.5 Schema del nucleo idrofobico

2.3 La modellazione su reticolo

Per inserire la struttura proteica nel reticolo, ovvero per eseguire l'embedding della sequenza di amminoacidi H e P, si hanno tre metodi per poterlo effettuare [13]:

1. *Coordinate Cartesiane :* la posizione dei residui è specificata in maniera indipendente dagli altri residui;
2. *Coordinate Interne :* la posizione di ogni residuo dipende dal suo predecessore nella sequenza di residui. Ci sono due tipi di coordinate interne : *assolute* dove le direzioni dei residui sono relative agli assi definite dal reticolo, *relative* dove le direzioni dei residui sono relative alla direzione della mossa precedente;
3. *Matrice Distanza :* la locazione di un dato residuo è calcolata attraverso la propria matrice distanza.

Krasnogor [14] fece un studio comparativo esaustivo usando algoritmi evoluzionistici con coordinate interne assolute e relative. L'esperimento mostrò che le coordinate relative sono più performanti rispetto le assolute nei reticoli quadrati e cubici, mentre per reticoli triangolari sono migliori le assolute.

Tuttavia, in generale, è difficile valutare l'efficacia della codifica usata sulle performace degli algoritmi dipende dalle modifiche che si intendono eseguire alla conformazione. In questa tesi, concordando con le affermazioni di Krasnogor, viene usato un reticolo cubico con coordinate relative.

Capitolo 3

Termodinamica del Protein Folding nel modello HP

L'adozione del modello HP, oltre che a presentare semplificazioni che riguardano la struttura della conformazione della proteina, influisce e semplifica anche le ipotesi della termodinamica che regolano il ripiegamento proteico stesso.

I concetti di energia, entropia, entalpia ed energia libera per il protein folding usando un modello HP sono stati esposti nel lavoro di Rainer König e Thomas Dandekar [15]. Le ipotesi iniziali sono quelle che si sono adottate anche in questo lavoro di tesi, ovvero il processo di ripiegamento avviene in un ambiente circondato da soluzione acquosa e la funzione energia è valutata solamente considerando l'energia idrofobica.

3.1 Entropia del Solvente

Con il termine entropia del solvente (*entropy solvent*), che viene utilizzato nel paper di R.König e T.Dandekar [15], si identifica una forza che agisce sul processo di ripiegamento e sul design della proteina ma che risulta abbastanza difficoltoso la sua modellazione.

L'implementazione del solvente e specialmente della propria entropia non si trova spesso in letteratura. Il calcolo esatto dell'entropia è difficile in sistemi reali. Calcolare il numero di microstati solventi sottolinea una differente prospettiva del ripiegamento proteico e della stabilità della proteina: il minimo globale non appare più come uno stato particolare tra tutte le possibili conformazioni, ma come la conformazione proteica con il più alto numero di rappresentazione dei microstati del solvente. L'entropia del solvente è dovuta dalla soluzione acquosa dove è immersa la proteina; tuttavia l'enumerazione dei differenti microstati[4] può essere difficoltosa.

Nel lavoro di Rainer König e Thomas Dandekar [15] è presentata una interessante implementazione dell'entropia solvente nel contesto del modello HP.

4 Un **microstato** descrive una specifica e dettagliata configurazione di un sistema, che il sistema attraversa durante le sue fluttuazioni termiche.

I risultati del lavoro indicano che considerare anche l'azione dell'entropia, oltre che a quelle dell'energia, nel modello HP, può portare ad una migliore predizione della struttura proteica, e in questo caso, del nucleo idrofobico.

Nel modello tridimensionale ogni punto della catena può girare a sinistra, destra, avanti, indietro, su e giù; ma la costruzione della catena è simile al modello bidimensionale, così come spiegato nel lavoro in questione [15].

3.2 Funzione di Fitness

Data una sequenza iniziale ($A_1, A_2, \ldots, A_n$) su un alfabeto {H,P}, l'obiettivo è trovare un struttura ripiegata S =[(x_1, y_1, z_1) … (x_n, y_n, z_n)] con il più basso valore possibile di Free/Global energy in uno spazio reticolare 3D, dove (x_i, y_i, z_i) corrisponde alle coordinate cartesiane dell'i-esimo residuo della sequenza.

La struttura S è valida, se e solo se sono rispettate le seguenti condizioni :

- per ogni $1 \leq i \neq j \leq n$, $(x_i, y_i, z_i) \neq (x_j, y_j, z_j)$
- per ogni $1 \leq i \leq n-1$, $|(x_{i+1} - x_i) + (y_{i+1} - y_i) + (z_{i+1} - z_i)| = 1$
- per ogni $1 \leq i \neq j \leq n$, $x_i, y_i, z_i \in R$

Normalmente l'energia libera (FE) è il valore negativo del numero di contatti di vicini idrofobici nello spazio, che non sono consecutivi nella sequenza HP iniziale. Il maggior svantaggio di questa definizione energetica è che non prende in considerazione le interazioni di H-residui che non sono direttamente adiacenti nello spazio.

Per ovviare a questo problema, I.Berenboym e M.Avigal[16], definiscono una nuova funzione energia, chiamata Global Energy (GE).

La definizione formale di entrambe le energie è data dalle equazioni seguenti :

$$FS(S) = -\sum_{i=1}^{n-2}\sum_{j=i+2}^{n} W_{ij} \quad , GE(S) = -\sum_{i=1}^{n-2}\sum_{j=i+2}^{n} W'_{ij}$$

$$\mathbf{W_{ij}} = \begin{cases} 1, se\ A_i = A_j = 'H'\ e\ |x_i - x_j| + |y_i - y_j| + |z_i - z_j| = 1 \\ 0, altrimenti \end{cases}$$

$$\mathbf{W'_{ij}} = \begin{cases} \dfrac{1}{|x_i - x_j|^2 + |y_i - y_j|^2 + |z_i - z_j|^2}, se\ A_i = A_j = 'H' \\ 0, altrimenti \end{cases}$$

A_i rappresenta l' i-esimo residuo della sequenza e può assumere il valore 'H' o 'P', W_{ij} è il peso di una interazione tra i residui i ed j, e (x_j, y_j, z_j) sono le coordinate dell'i-esimo residuo della struttura S nello spazio 3D.

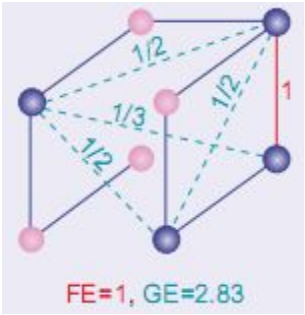


Figura 3.1 Reticolo cubico e contatti idrofobici Global Energy

3.3 Calcolo dell'entropia e dell'entalpia

Per studiare gli effetti dell'entropia, immaginiamo di riempire tutte le posizioni vuote nel reticolo con delle molecole di acqua. Le interazioni a lungo raggio del solvente, ovvero quelle che non riguardano a posizioni adiacenti, non vengono considerate in questo semplice modello. I piccoli complessi acquosi circondano il modello della proteina con due differenti proprietà : ordinati e meno ordinati. I complessi acquosi ordinati sono adiacenti ai residui idrofobici o al solvente stesso. I complessi meno ordinati (ad alta entropia) esistono se nessun residuo idrofobico è adiacente ad essi. Un O-blocco (ordinato) sul reticolo è assegnato al complesso acquoso ordinato, un L-blocco è un complesso poco ordinato.

Nella figura 3.2 è riportato quanto appena spiegato.

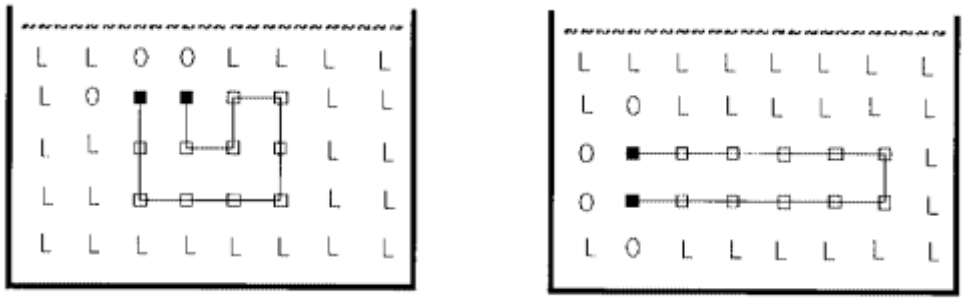


Figura 3.2 L-blocchi O-blocchi

Il numero di complessi acquosi non ordinati, è N_i. Sperimentalmente è stato dimostrato che le molecole idrofobiche riducono l'entropia del solvente acquoso che circonda la proteina.

Diversamente, la semplice funzione energia non differenzia tra insiemi ordinati e non ordinati.

Il modello permette lo studio di tre implementazioni : l'energia (ΔE, semplificata nel contesto del modello alla sola considerazione dei contatti idrofobici), l'entropia (ΔS, differenza di insiemi acquosi ordinati) e la combinazione di esse, F = E - TS. F indica l'energia libera di Helmholtz[5](corrisponde all'energia libera di Gibbs se si trascurano i cambiamenti di temperatura e pressione).

Il valore dell'energia E viene calcolato tramite la funzione di fitness. Mentre l'entropia S di una proteina su reticolo semplificata col modello HP si calcola semplicemente contando gli insiemi acquosi ordinati. Ottenuto ciò si può procedere al calcolo dell'energia libera di Helmholtz, che sotto queste ipotesi corrisponde anche all'entalpia della proteina [15].

Il calcolo di questi tre valori sarà utile per la descrizione dell'energy landscape associato alle configurazioni tridimensionali della proteina. Inoltre, cercando di minimizzare i valori di energia e di entropia, si otterrà una ottimizzazione della struttura del nucleo idrofobico. Questo perché le configurazioni a minor energia ed entropia sono quelle con il nucleo idrofobico più compatto e con pochi contatti con la soluzione acquosa che accerchia la proteina.

5 L'**energia libera di Helmholtz** è una funzione di stato utilizzata in termodinamica per rappresentare l'energia libera nelle trasformazioni a volume costante. Prende il nome dal fisico e fisiologo tedesco Hermann von Helmhotz.

3.4 Il parametro temperatura

Nello studio termodinamico della modellazione proposta, un parametro che più volte appare nelle formule di calcolo è la temperatura del processo di folding. La temperatura nel processo di folding è molto importante, infatti, è dimostrato[17] che ad alte temperatura la proteina non si porta allo stato nativo, ma "preferisce" stare nei random coil[6], che sono stati ad alta entropia; mentre a temperature basse, il processo di folding è più lento perché non riesce a saltare le barriere di potenziale energetico e quindi i minimi locali, per convergere allo stato nativo. Detto ciò, si definisce temperatura ottimale, la temperatura T che permette il ripiegamento proteico nel minor tempo possibile. Il valore di questo parametro è ottenibile solo per via empirica, in base alla proteina da studiare. Data l'impossibilità di avere valori esatti della temperatura nella simulazione su computer, si ha la necessità di avere un range di valori per la temperatura, questo parametro viene detto temperatura di simulazione. Nella letteratura scientifica presente [18, 19], sono riportati i range ritenuti ottimali per una simulazione del ripiegamento su reticolo; tali valori sono racchiusi tra 2 e 5. La distribuzione di questi valori segue le leggi della termodinamica, ovvero, più ci si avvicina allo stato nativo, e minore è il valore della temperatura.

6 Il **random coil** è l'effetto dell'azione di agenti denaturanti sull'organizzazione strutturale delle proteine native. Sono strutture a "caso".
Queste sostanze agiscono labilizzando le interazioni non covalenti esistenti nella proteina e generando una struttura disorganizzata ed estremamente flessibile che, variando continuamente e casualmente la sua conformazione in soluzione, descrive un gomitolo virtuale (gomitolo statistico). In queste condizioni la proteina è totalmente svolta e denaturata e perde qualsiasi funzione biologica.
Questa condizione, che si ottiene solo in laboratorio, serve come stato di riferimento certo in esperimenti che studiano le caratteristiche strutturali e/o termodinamiche delle proteine. Poiché la proteina è un eteropolimero e le soluzioni impiegate per denaturarla non sono del tutto "ideali" (nel senso di labilizzare totalmente ed efficacemente "tutte" le interazioni non covalenti), talvolta nelle soluzioni di proteina denaturata si riscontra (mediante tecniche spettroscopiche) una organizzazione strutturale residua.

Capitolo 4
L'algoritmo genetico

4.1 Introduzione

L'algoritmo genetico (GA), appartiene alla categoria degli algoritmi evoluzionistici, e fu proposto da John Holland [20] (*University* of *Michigan*) tra gli anni 1960-1970. Il paradigma di elaborazione delle informazioni che racchiude questi algoritmi è detto, *Computazione Evoluzionistica*. Tale paradigma è, da sempre, praticato e sviluppato nei processi naturali; e successivamente, quando l'uomo è riuscito a individuarlo, è stato modellato il suo principio generale per algoritmi computazionali per essere usati nei computer.

L'algoritmo genetico modella quindi, il processo di evoluzione presente in natura, che è così brevemente definito: *"L'**evoluzione** è il fenomeno di cambiamento, attraverso successive generazioni, del patrimonio genetico delle specie (il genotipo) e conseguentemente della sua manifestazione somatica (il fenotipo).*

Tale processo si basa sulla trasmissione del patrimonio genico di un individuo alla sua progenie e sull'interferenza in essa frapposta dalle mutazioni casuali. Sebbene i cambiamenti tra una generazione e l'altra siano generalmente piccoli, il loro accumularsi nel tempo può portare un cambiamento sostanziale nella popolazione, attraverso i fenomeni di selezione naturale e deriva genetica, fino all'emergenza di nuove specie." [20, I1]

Rispetto alla teoria evoluzionista della Biologia, c'è uno scostamento concettuale che riguarda i motivi dell'evoluzione. In ambito biologico l'evoluzione è determinata dall'habitat in cui vive la popolazione, e quindi non necessariamente le nuove popolazione saranno "migliori" ma solamente più "adatte". Mentre, per quanto riguarda la sua modellazione algoritmica, l'evoluzione seleziona i migliori individui, fino ad arrivare all'individuo ottimo. L'evoluzione dunque ha il fine di ottimizzare una funzione di *fitness* (testa la qualità degli individui della popolazione) e di generare nuove popolazioni migliori delle precedenti.

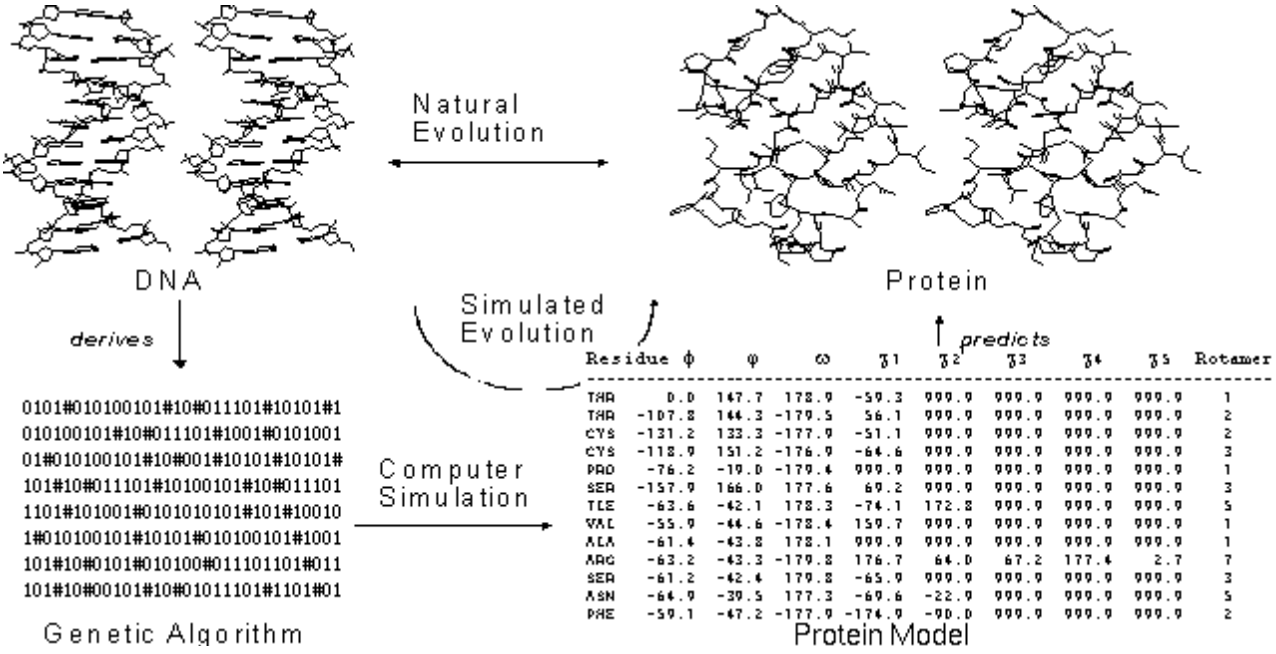


Figura 4.1 Schema Evoluzione Naturale e Evoluzione Simulata

Tutti i metodi che appartengono alla Computazione Evoluzionistica, sono metodi euristici. Quindi non possono garantire di trovare né la soluzione ottima, né di essere adatti alla soluzione di qualsiasi problema.

Riassumendo, un algoritmo genetico è un algoritmo di ricerca iterativa il cui scopo è l'ottimizzazione della funzione di fitness. Partendo da una popolazione iniziale, un algoritmo genetico produce nuove generazioni che contengono (di solito) individui migliori delle precedenti, in altre parole, l'algoritmo evolve verso l'ottimo globale della funzione di fitness.

Come detto in precedenza, il problema del ripiegamento proteico è di tipo NP, e quindi utilizzo di un algoritmo genetico (o le euristiche in generale) risulta una della opzioni più promettenti per giungere ad una soluzione. Inoltre l'algoritmo genetico lascia la possibilità di essere adattato e integrato con altre tecniche nei vari passi della sua implementazione, a seconda delle peculiarità che il problema presenta. In questo lavoro di tesi si è voluto sfruttare questa possibilità di "personalizzazione" dell'algoritmo, modificando la fase di selezione.

4.2 La codifica

La codifica genetica è il procedimento che identifica le soluzioni del problema nei cromosomi artificiali. In altri termini si deve definire cosa rappresenta il genotipo e il fenotipo nella risoluzione del problema. Ogni individuo viene quindi codificato tramite un cromosoma (genotipo), che generalmente è una stringa (binario o non).

Formalmente la funzione di codifica può essere così definita :

$C : S \rightarrow X$,

dove S è lo spazio delle soluzioni del problema ed X è lo spazio dei geni (spazio di ricerca).

Dato $c \in C(s)$ *e,* $s \in S$ *,* la soluzione s può essere interpretata come il fenotipo associato al genotipo c [21].

4.3 La funzione di fitness

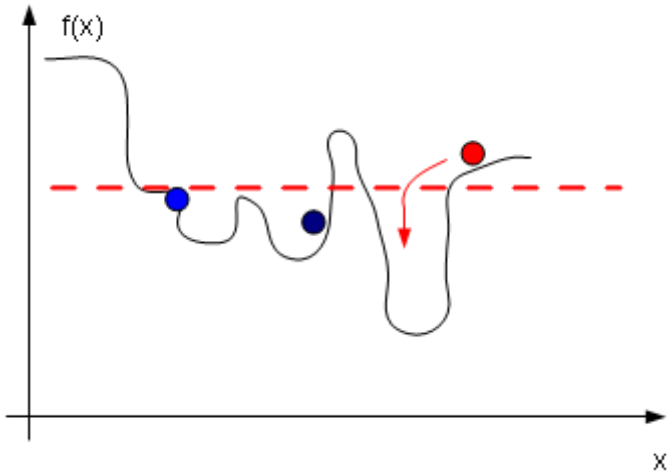
Definita la codifica del gene, rimane da definire la funzione di fitness. Questa funzione è la funzione da minimizzare o massimizzare, essa descrive la qualità di ogni individuo. Generalmente la funzione ritorna un valore numerico, che serve per effettuare una classifica della popolazione in base alla qualità, inoltre questo parametro è normalmente considerato nella fase di selezione.

4.4 La selezione

Il primo passo per la soluzione è la generazione di un numero di cromosomi (popolazione) in maniera random, i quali avranno funzione di fitness molto bassa. Fatto ciò, i passi successivi dell'algoritmo consistono nell'applicazione di alcune operazioni, che tendono a modificare la popolazione di geni, nel tentativo di migliorarli in modo da ottenere una soluzione sempre migliore (ottima o quasi).

L'evoluzione procede iterando i vari passi dell'algoritmo, per ogni iterazione viene per

prima cosa eseguita una selezione dei geni, in base ad opportune tecniche di selezione. Con il processo di selezione, l'algoritmo intende prelevare gli individui più promettenti dai quali generare la nuova popolazione. Non sempre per la riproduzione della popolazione si selezionano gli individui che ottimizzano la funzione di fitness, questo perché comporterebbe una convergenza verso un minimo locale come si mostra in figura 4.2.



La selezione nell'esempio qui a fianco avviene solamente basandosi sul valore della funzione di fitness.
In questo caso il valore di selezione è delimitato dalla linea tratteggiata. Con questo tipo di selezione però si scarta il punto rosso, che ci permetterebbe di arrivare al minimo globale in minor tempo.

Figura 4.2 Esempio di Selezione errata

Il processo di selezione oltre a cercare di dare una rapida convergenza all'algoritmo deve anche garantire di non convergere rapidamente in minimi (o massimi) locali, creando così a perdita del patrimonio genetico per riuscire ad uscire dall'eventuale ottimo locale.

La strategia di selezione appare chiaramente è uno dei punti cruciali dell'algoritmo genetico. Se si selezionano individui con alto fitness ma simili tra loro, avremo una soluzione in tempi rapidi, ma probabilmente non ottima, in quanto solo una parte dello spazio delle soluzioni sarà esplorato. Di contro, se tentiamo di mantenere in fase di selezione, molti individui differenti tra loro ma con funzioni di fitness basse, avremmo una soluzione più vicina all'ottimo globale, ma in tempi decisamente maggiori.

4.5 Gli operatori genetici : crossover e mutazione

L'algoritmo, successivamente alla fase di selezione, consiste nell'applicazione di operazioni, che tendono a modificare la popolazione dei geni. Queste operazioni sono applicate su un numero di cromosomi stabilito, che in generale determinano quanti cromosomi devono subire crossover e mutazioni, e in quale misura.

Il crossover prende due individui, messi nell'insieme “mating pool” dal processo di selezione, e taglia le stringhe dei loro due cromosomi in qualche posizione scelta a caso, per produrre due segmenti "testa" e due segmenti "coda" (tail). I segmenti coda sono poi scambiati per produrre due nuovi cromosomi di lunghezza completa. Ciascuno dei figli eredita alcuni geni da ogni genitore. Questo è conosciuto come single point crossover.

Una definizione formale del processo di crossover può essere la seguente,

siano $<A_1, A_2, \ldots, A_n>$ e $<B_1, B_2, \ldots, B_n>$ le due stringhe “generatrici”, e sia k ($1 \leq k \leq n$) il punto di crossover, il risultato di questa operazione saranno le seguenti stringhe figlie :

$<A_1, A_2, \ldots, A_k, B_{k+1}, \ldots, B_n>$ e $<B_1, B_2, \ldots, B_k, A_{k+1}, \ldots, A_n>$.

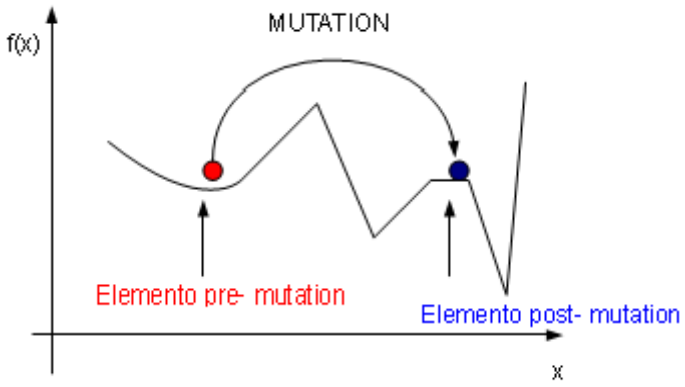
Questo è conosciuto come single point crossover.

Tuttavia, sono stati presenti diversi algoritmi di crossover, che spesso coinvolgono più di un punto di taglio (multiple crossover). Per esempio, questa è la definizione del double-point crossover : siano $<A_1, A_2, \ldots, A_n>$ e $<B_1, B_2, \ldots, B_n>$ le due stringhe “generatrici”, e siano k e l($1 \leq k, l \leq n$) i punti di crossover, il risultato di questa operazione saranno le seguenti stringhe figlie :

$<A_1, A_2, \ldots, A_k, B_{k+1}, \ldots, B_l, A_{l+1}, \ldots, A_n>$

$<B_1, B_2, \ldots, B_k, A_{k+1}, \ldots, A_l, B_{l+1}, \ldots, B_n>$.

Il crossover è un'operazione binaria, in quanto necessita di due individui per essere eseguita, mentre la mutazione è un'operazione unaria. La mutazione è applicata a ogni individuo singolarmente, che viene alterato con una data probabilità. La modifica del gene, serve per mantenere un patrimonio genetico della popolazione abbastanza vario, in altre parole è utile per effettuare i “salti” per fuggire dalla convergenza nell'ottimo locale.



L'operatore Mutazione permette di variare le caratteristiche dei cromosomi che compongono la popolazione dell'algoritmo genetico, permettendo così una esplorazione dello spazio più completa per la ricerca del minimo globale.

Figura 4.4 Effetto dell'operatore mutazione

La teoria tradizionale ritiene che il crossover sia più importante della mutazione per quanto riguarda la rapidità nell'esplorare lo spazio di ricerca. La mutazione porta un po' di "casualità" nella ricerca e aiuta ad assicurarci che nessun punto nello spazio abbia probabilità nulla di essere esaminato, inoltre previene la deriva genetica ovvero il convergere dei membri della popolazione verso qualche punto dello spazio di ricerca. Questo è dovuto al fatto che un gene predominante si può propagare a tutta la popolazione. Una volta che un gene converge in questa maniera il crossover non può introdurre nuovi valori. Da questo si evince che mentre la popolazione si avvicina alla convergenza la mutazione diventa più produttiva del crossover.

Dopo che i figli sono stati prodotti attraverso la selezione, il crossover e la mutazione degli individui della vecchia generazione, bisogna calcolare il loro fitness e reinserirli nella popolazione. A questo punto si hanno due possibilità: global reinsertion o local reinsertion.

A loro volta, due tecniche sono divise in differenti possibili implementazioni, per esempio la global reinsertion può avvenire nelle seguenti maniere:

- *pure reinsertion*, la vecchia popolazione viene sostituita integralmente;
- *uniform reinsertion*, vengono prodotti figli in numero inferiore ai genitori che vengono rimpiazzati in maniera uniforme e casuale;
- *elitist reinsertion*, vengono rimpiazzati i genitori peggiori;
- *fitness-based reinsertion*, viene generata una prole maggiore di quella richiesta e vengono reinseriti solo i migliori individui della prole;

Mente nella local selection gli individui vengono selezionati da un insieme limitato e contiguo. L'inserimento avviene esattamente nello stesso insieme, in questo modo viene preservata la località dell'informazione. Per la selezione di un genitore da rimpiazzare e per

quella di un figlio da inserire viene seguito uno di questi schemi:

- Tutti i figli vengono inseriti nell'insieme e gli individui vengono rimpiazzati in modo casuale;
- Tutti i figli vengono inseriti nell'insieme e gli individui peggiori vengono rimpiazzati;
- I figli migliori vanno a sostituire gli individui peggiori nell'insieme;
- I figli migliori prendono il posto dei genitori nell'insieme;
- I figli migliori vanno a sostituire individui scelti a caso nell'insieme;
- I genitori vengono rimpiazzati dai figli migliori.

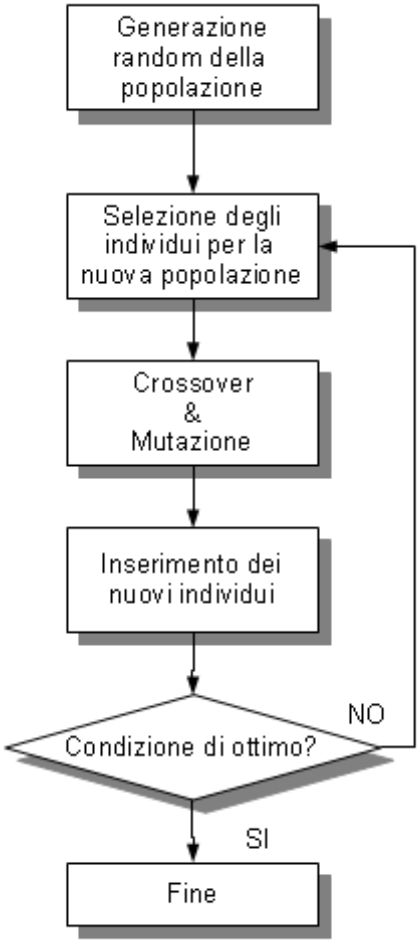


Figura 4.5 Schema Algoritmo Genetico

Capitolo 5
L'energy landscape

5.1 Introduzione

In questo lavoro di tesi, si vuole sfruttare lo studio dell'energy landscape per poter migliorare processo di selezione degli individui dell'algoritmo genetico, così facendo si cerca di dare una convergenza all'algoritmo più rapida.

Lo studio sullo spazio energetico nasce fondamentale dalla necessità di comprendere meglio il ripiegamento delle proteine e l'energy landscape ad esse associato. Perché come riportato in molti lavori sulla predizione proteica [22, 23, 24] la comprensione dell'energy landscape fornisce un'idea su come sviluppare algoritmi di previsione della struttura.

Nel corso del capitolo, viene spiegato cosa si intende per spazio energetico e come esso è definito. Inoltre è definito il passaggio dalla configurazione tridimensionale su reticolo della proteina all'energy landscape ad essa associato (spazio energetico).

5.2 Lo spazio energetico

In fisica, un *energy landscape* (paesaggio energetico) è una coppia (X,f) costituita da uno spazio topologico X, il quale rappresenta gli stati o i parametri fisici di un sistema, e la funzione continua $f : X \to \mathbf{R}^n$, che rappresenta l'energia associata agli stati o parametri che l'immagine di f rappresenta una ipersuperficie in $\mathbf{R}^n$.

Il minimo di un energy landscape rappresenta il punto più stabile si tutto il sistema termodinamico.

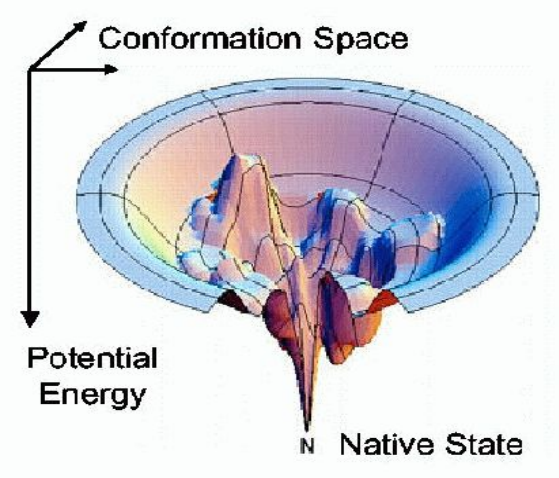


Figura 5.1 Energy Landscape

Il dogma di Levinthal [cap.1, par.5] afferma che, la superficie di energia libera nelle vicinanze dello stato nativo deve essere piuttosto ripida ed elevata, per potere fornire la stabilità. Così descritta la superficie di energia libera è simile ad un campo da golf. Questo perché siamo nelle condizioni descritte da Levinthal sono quelle del paradosso, ovvero che la proteina prima di ripiegarsi deve passare in tutte le configurazioni possibili.

Figura 5.2 Funnel Energy

Nel corso degli anni '80, fino all'inizio degli anni '90, Joseph Bryngelson e Peter Wolynes hanno lavorato sulla formulazione della teoria dell'energy landscape, riguardo il protein folding [25].

Questo approccio ha introdotto il *principio di frustrazione minimale*, che afferma che

l'energy landscape relativo all'evoluzione del processo di folding di una proteina presenta dei "canali" che vanno verso lo stato nativo della proteina. Ne risulterebbe un energy landscape a "funnel" (a imbuto), figura 5.3, che consente il ripiegamento della proteina allo stato nativo tramite qualunque di uno dei percorsi intermedi che convergono allo stato nativo, piuttosto che essere limitata ad un unico meccanismo. Ovviamente però, solamente uno sarà il percorso ottimo.

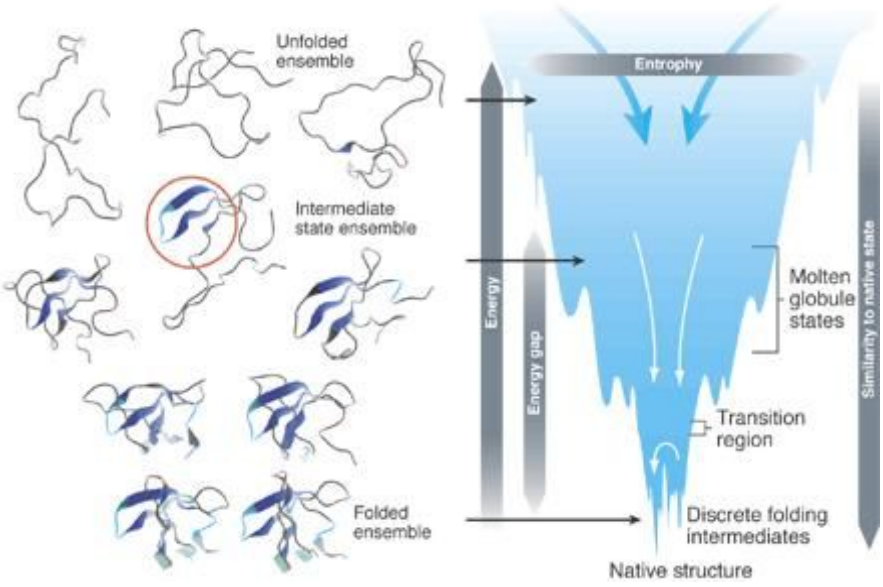


Figura 5.3 Funnel Energy del protein folding

Basandosi su questa teoria, si è pensato di sfruttare lo studio dell'energy landscape in fase di selezione dell'algoritmo genetico, in maniera di poter effettuare una scelta della popolazione da riprodurre più accurata e in modo tale da poter arrivare alla configurazione nativa con maggior rapidità.

Studi teorici hanno dimostrato [22, 26, 27] come le superficie di energia potenziale fatte ad imbuto con unico mimino possano guidare efficientemente una proteina verso strutture native grazie alla progressiva organizzazione delle strutture parzialmente ripiegate che si formano lungo il cammino.

Questo dal punto di vista strettamente teorico, nella realtà la forma ad imbuto non è così semplice e "liscia" da portare allo stato nativo in maniera rapida. Il problema dei corrugamenti dell'energy landscape è da considerare nello studio del landscape relativi alle proteine. Il fenomeno di corrugazione è dato da impedimenti locali (contatti non nativi, impedimenti sterici, ecc...) che producono barriere di potenziale alcune volte maggiori delle

fluttuazioni termiche. In altre parole, nell'energy landscape sono presenti delle barriere di potenziale che causano delle increspature su esso. Durante il ripiegamento, questi corrugamenti sull'imbuto influisce sul processo di folding intrappolando alcune configurazioni all'interno di queste barriere di potenziale.

Per descrivere un energy landscape, si necessita delle coordinate [28], se quella relativa alla profondità appare ovvia, l'energia idrofobica; più delicato l'individuazione delle coordinate di reazione. Dalla definizione che la Meccanica Classica da di coordinate di reazione, si può dire che sono le variabili rilevanti di un processo fisico e nelle quali si possa descrivere il pontenziale termodinamico.

Per descrivere completamente la termodinamica del protein folding, appare quindi evidente che oltre all'energia idrofobica, si necessita sia dell'entropia[7] che dell'energia libera[8] del processo.

Nell'energy landscape, ogni punto è una configurazione della proteina (configurazione nello spazio delle conformazioni [22]) con associata la propria energia. Per differenziare lo spazio dove si sviluppa l'energy landscape, definito dall'energia libera e dalle coordinate di reazione, da quello delle conformazioni, lo chiameremo, spazio *energetico.*

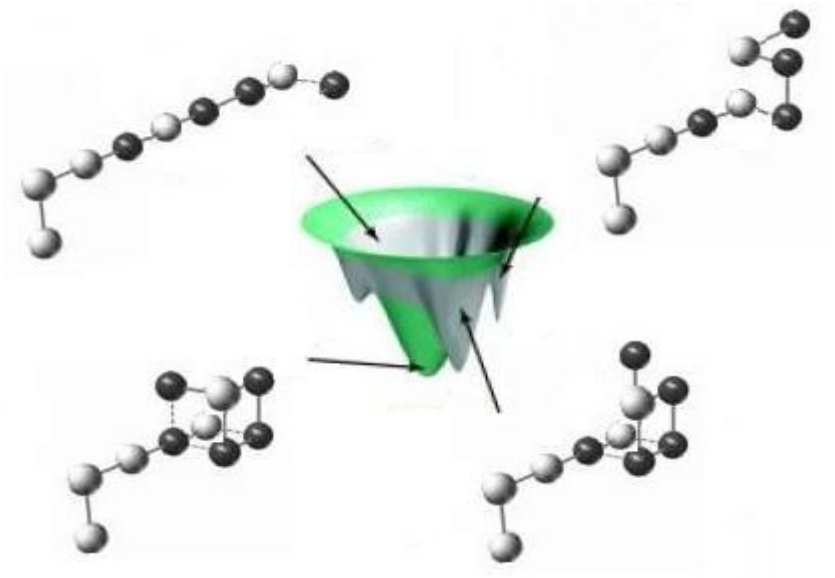


Figura 5.4 Spazio delle configurazione – Spazio Energetico

7 In fisica l'**entropia** è una grandezza che viene interpretata come una misura del caos di un sistema fisico o più in generale dell'universo. Viene generalmente rappresentata dalla lettera *S*.

8 L'**entalpia**, solitamente indicata con *H*, è una funzione di stato che esprime la quantità di energia che un sistema termodinamico può scambiare con l'ambiente. L'entalpia è definita dalla somma dell'energia interna e del prodotto tra volume e pressione di un sistema.

5.2Lo spazio energetico

Quindi l'energy landscape contiene tutte le conformazioni della proteina e l'energia ad esse associata. La funzione (la funzione φ riportata in figura 5.5) che mappa una configurazione dello spazio delle configurazioni in un punto nello spazio energetico non è biunivoca, in quanto vi possono essere più conformazioni che mappano lo stesso punto. Questo può essere, o perché si considerano configurazioni speculari tra loro, o pur essendo strutture diverse hanno stesse coordinate. Tuttavia, in entrambi i casi non ci sono complicanze concettuali, perché uno studio effettivo sulla similarità delle strutture non viene eseguito in questa tesi, e quindi dal punto di vista dell'algoritmo genetico tutto ciò non ha rilevanza.

La gestione di questo caso particolare non deve essere specifica nell'algoritmo genetico .

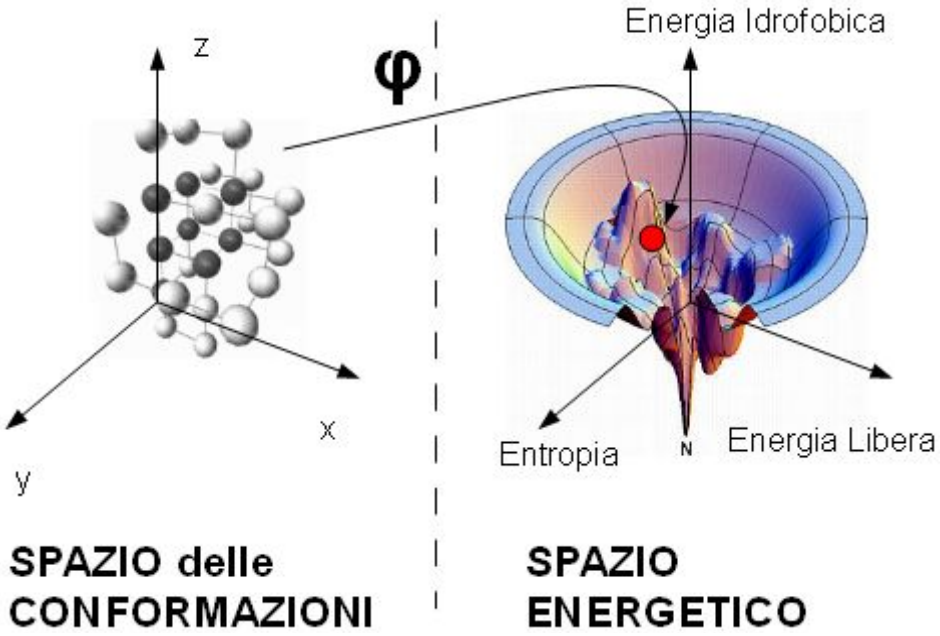


Figura 5.5 Funzione mappatura

Nel processo di folding, la configurazione della proteina cambia molte volte. Ogni volta che una configurazione cambia (effettua una transizione) ad una configurazione nuova ma similare, si ha anche una transizione da un punto ad un altro anche nell'energy landscape. Tuttavia, una volta definito l'energy landscape, si può calcolare la probabilità di transizione da un punto ad un altro all'interno dell'energy landscape, e simulare così il processo di folding.

5.3 Transizioni probalistiche sull' Energy Landscape

5.3.1 Markov Model of Transitions

Il processo di folding può essere considerato come un processo di transizioni probabilistiche nell'energy landscape. Questo processo probabilistico viene eseguito su un modello di Markov, dove la probabilità di andare nello stato successivo solo dipende dallo stato corrente. In altre parole, la transizione tra due configurazioni è statica e solo dipende sull'energy landscape ma non dipendono dallo stato precedente del processo di transizione. Così, l'energy landscape può essere modellato come una rete di transizione di Markov, dove ogni nodo è una configurazione della proteina, mentre la probabilità di transizione tra vicini è la probabilità di transizione di Boltzmann [24, 29].

5.3.2 Transition Probability

Ci sono molte regole per calcolare la probabilità di transizione di Boltzmann, in questo lavoro si è scelto di usare la regola di Metropolis per il calcolo della probabilità.

La probabilità di transizione K_{ij} risulta essere :

$$K_{ij} = \begin{cases} e^{\left(\frac{-\Delta E}{kT}\right)} \; se \; \Delta E > 0 \\ 1 \; se \; \Delta E < 0 \end{cases}$$

con $\Delta E = E_j - E_i$, k è la costante di Boltzmann e T è la temperatura del folding.

5.3.3 Boltzmann Equilibrium Distribution

Le transizioni tra configurazioni finirà per stabilizzarsi e raggiungere l'equilibrio dove la popolazione delle configurazioni non cambierà, ovvero arriverà al minimo dell'energy landscape.

Distribuzione di equilibrio del protein folding può essere calcolati a partire dalla energia libera E di ogni con figurazione. Il fattore di distribuzione di Boltzmann P_i di una data conifigurazione con energia libera E_i è : $P_i = E$;

dove k è la costante di Boltzmann[9] e T è la temperatura del folding.

9 Il parametro k, costante di Boltazmann, può essere per semplicità posto a 1. Questo solo se si è in presenza di una modellazione proteica fatta con il modello HP e nell'ipotesi che la proteina sia circondata da una soluzione acquosa. Riferimenti in merito a questa ulteriori semplificazione sono reperibile nel lavoro [15].

5.3.4 Detailed Balance

Le probabilità di transizione tra due configurazione P_i e P_j dovrebbe soddisfare il detailed balance in modo che nella distribuzione di equilibrio , sia bilanciata per entrambe le direzioni della transizione : $P_i \times K_{ij} = P_j \times K_{ji}$.

Nella condizione di equilibrio le configurazioni dovrebbero stare nella distribuzione di Boltzmann, quindi le probabilità di transizione dovrebber soddisfare il detailed balance, cioè :

$$\frac{K_{ij}}{K_{ji}} = e^{(\frac{-(E_j - E_i)}{kT})}$$

L'equazione di Metropolis vista precedentemente, soddisfa il detailed balance.

5.4 Probabilist Roadmap Methods

I metodi Probabilistic Roadmap (PRM) sono metodi random per risolvere problemi di motion planning. Sono quindi usati per il movimento di robot nello spazio con ostacoli, e dato il punto di partenza e di arrivo, questo metodo trova il percorso ottimo senza collisioni con gli ostacoli [30].

Brevemente, verrà illustrato il funzionamento di questi metodi. Il PRM lavora campionando dei punti casualmente nel C-space e vengono mantenuti quelli che non collidono con i vincoli di fattibilità. Il PRM lavora sullo spazio generale C-space, che a sua volta è diviso in due tipi di spazio : C-obstacle e C-free. C-obstacle è lo spazio occupato degli ostacoli, mentre C-free è lo spazio dove non ci sono gli ostacoli.

Dopo aver campionato il C-space e selezionato solo i punti presenti dentro lo spazio C-free, si procede alla connessione dei punti come in un grafo, connettendo tutti i punti vicini tra essi.

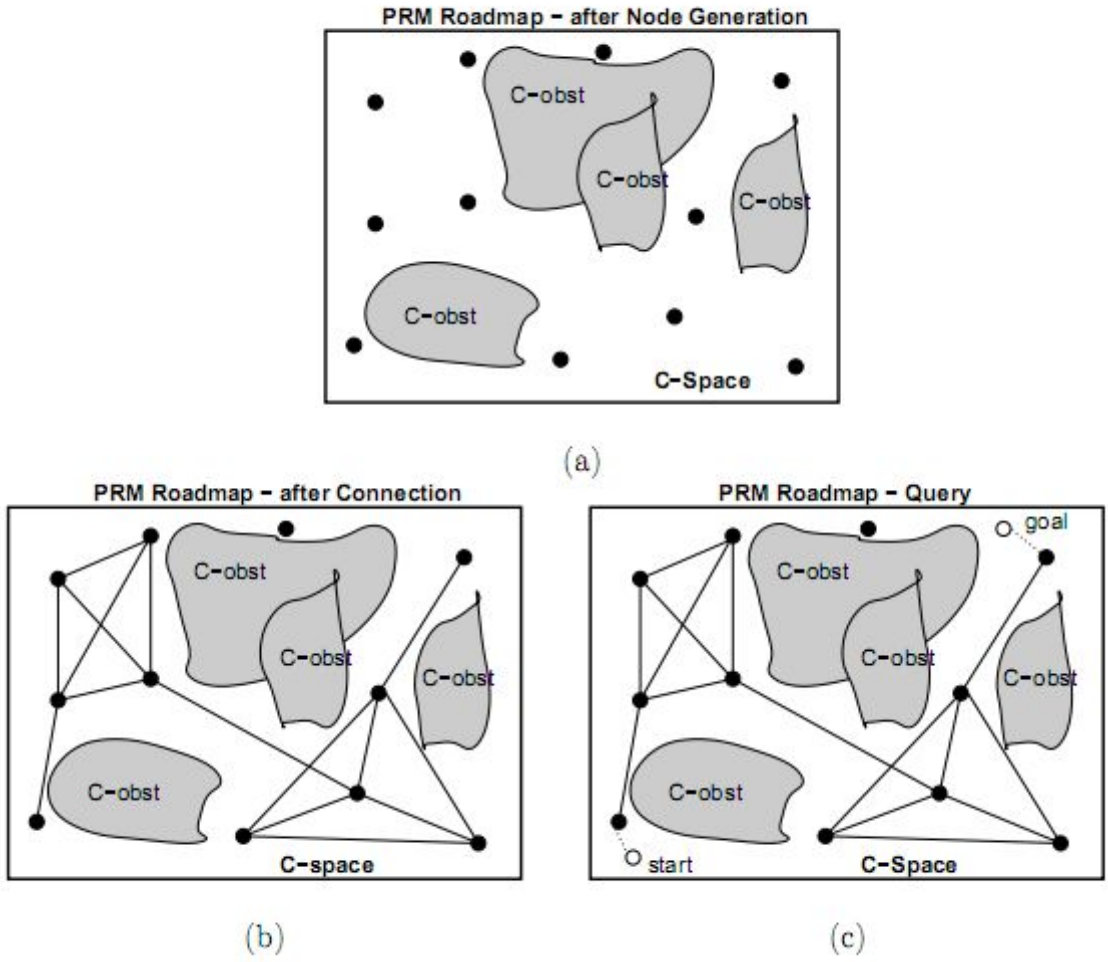


Figura 5.6 Processo del metodo PRM

5.5 Metodo Probabilistic Roadmaps per il Protein Folding

Una delle applicazione delle Probabilistic Roadmaps è riportata nella tesi di Tang [29], dove vengono usate per individuare il percorso di ripiegamento proteico ottimale. L'idea è quella di considerare la proteina in movimento verso lo stato nativo e le conformazioni non valide sono gli ostacoli di questo movimento. Il metodo come input ha lo stato iniziale (denaturato) e lo stato finale (nativo) della proteina ed esegue uno studio del processo di folding, e dei percorsi più probabili che la proteina segue per il ripiegamento.

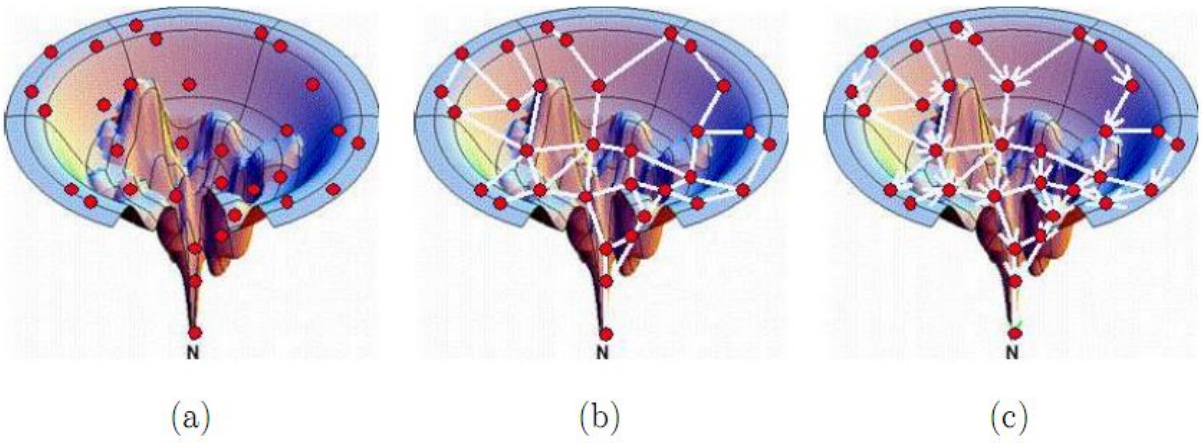


Figura 5.7 Folding PRM

Il metodo è riportato in figura 5.6, ed è formato in tre passi :

(a) campionamento dell'energy landscape;

(b) connessione tra i nodi campionati;

(c) analisi dell'energy landscape e dei percorsi descritti.

Un arco di connessione tra due nodi q_i e q_j, consiste in un arco pesato che rifletta la fattibilità energetica della transizione tra i nodi.

Successivamente a questa fase, ci sarà un algoritmo di ricerca del miglio percorso fattibile per la proteina.

Pseudo codice del Probabilistic Roadmaps per il protein folding:

I. Preprocessing: Costruzione della Roadmap
   1. Generazione dei nodi (trovare le configurazioni valide)
   2. Connessioni (connettere i nodi per formare la roadmap)

II. Query Processing
   1. Connettere il punto di inizio e di fine
   2. Trovare il percorso migliore nella roadmap tra le connessioni dei nodi

5.6 Come costruire la Probabilistic Roadmap

Nel lavoro di tesi, l'energy landscape non è descritto come un “funnel landscape”, ma è una superficie a energia libera (FES, Free Energy Surface) [26], come mostrato in figura 5.8.

La scelta di considerare l'energy landscape come una superficie è dettata da motivazioni di maggior semplicità dei calcoli; in quanto a livello teorico grazie alle ipotesi di lavoro considerate, con le relative semplificazioni energetiche, l'unica differenza tra le due modellazioni riguardano le coordinate. Le coordinate che descrivono l'energy landscape sono energia idrofobica, entropia del solvente ed energia libera, però nel caso del funnel landscape si devono considerare le coordinate polari dell'entropia e dell'energia libera.

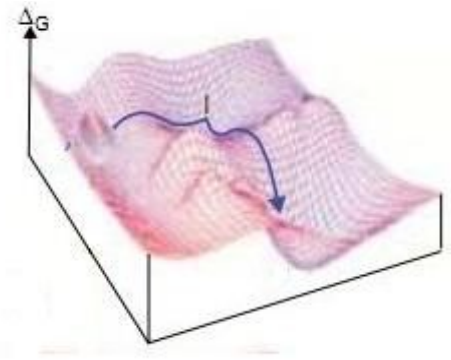


Figura 5.8 Energy Landscape

In questo lavoro di tesi, le probabilistic roadmaps sono utilizzate all'interno dell'algoritmo getico, in modo particolare viene usato nella fase di selezione degli individui ritenuti adatti per il crossover/mutation.

Ovviamente il metodo non è usato nella sua completezza, ma solamente in alcuni aspetti. Innanzitutto, non abbiamo a disposizione i punti di inizio e fine del percorso, in quanto il punto di fine è lo stato nativo (che è il risultato dell'algoritmo getico). Anche la fase di campionamento non è effettuata, perché i punti sono tutte le conformazioni relative agli individui della popolazione. La connessione tra i punti nell'energy landscape, invece, è la parte del metodo Probabilistic Roadmaps che viene implementata in questo lavoro. L'energy landscape è bidimensionale, in quanto si considerano solo le coordinate di entropia ed energia libera, questo perché l'informazione dell'energia libera è intrisicamente contenuto nel valore di energia libera.

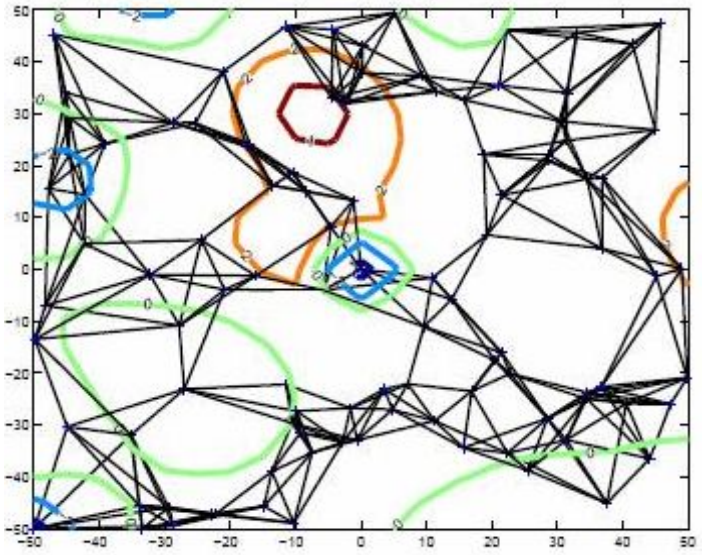


Figura 5.9 Energy Landscape

Una volta mappate le configurazioni nell'energy landscape, precedentemente descritto, si deve provvedere alla connessione dei punti. Tale processo avviene nel seguente modo :

- Si seleziona il punto ad energia libera minima;
- Si trovano i suoi k-vicini, attraverso l'algoritmo k-Nearest Neighbors;
- Si collegano i vicini con archi di peso pari alla probabilità di transizione di Boltzmann, la quale ci da la fattibilità energetica della transizione;
- Se l'arco ha peso inferiore al valore *w* (con *w* da stabilire), l'arco viene eliminato in quanto la transizione che descrive non viene considerata sufficientemente probabile;
- Si ripete il procedimento per tutti i nodi ancora non connessi.

5.6.1 L'algoritmo K-nearest neighbors

L'algoritmo k-nearest neighbors [31], è un metodo di classificazione basato sulla distanza degli oggetti. La parte relativa al classificazione non è importante, per lo scopo della tesi, ricopre un importante ruolo invece, la fase di partizione dello spazio (in questo caso, lo spazio energetico) e dell'individuazione dei vicini, dato un punto iniziale sullo spazio. Per vicini, si intente due punti a distanza euclidea “piccola”. Si presenta ora come l'algoritmo k-nearest neighbors.

Lo spazio viene partizionato in regioni in base alle posizioni e alle caratteristiche degli oggetti di training. Questo può essere considerato come il training-set per l'algoritmo anche se non è esplicitamente richiesto dalle condizioni iniziali.

Ai fini del calcolo della distanza euclidea gli oggetti sono rappresentati attraverso vettori di posizione in uno spazio multidimensionale. L'algoritmo lavora anche con altri tipi di distanza ad esempio la distanza Manhattan. L'algoritmo è sensibile alla struttura locale dei dati.

Un punto (che rappresenta un oggetto) è assegnato alla classe *C* se questa è la più frequente fra i k esempi più vicini all'oggetto sotto esame, la vicinanza si misura in base alla distanza fra punti. I vicini sono presi da un insieme di oggetti per cui è nota la classificazione corretta. Nel caso della regressione per il calcolo della media (classificazione) si usa il valore della proprietà considerata.

Il calcolo delle distanze è computazionalmente oneroso e proporzionale alla taglia dell'insieme di dati sotto esame. Gli algoritmi proposti che migliorano questo inconveniente cercano principalmente di diminuire il numero di distanze da calcolare per la decisione. In alcuni casi si cerca di partizionare lo spazio vettoriale e si calcolano solo le distanze tra volumi dello spazio vettoriale.

5.6.2 Kd-tree

La struttura dati che permette di effettuare una partizione dello spazio è il kd-tree. Il kd-tree è un caso particolare degli alberi BSP.

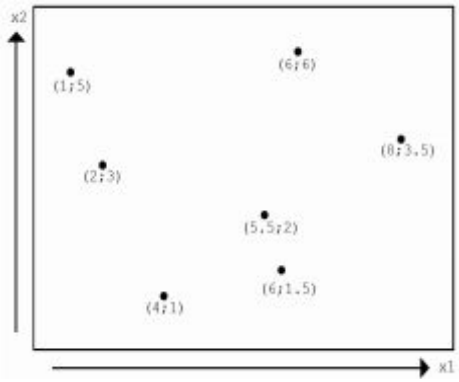
Il kd-tree è un albero binario in cui ogni nodo è un punto di k-dimensionale. Ogni nodo non foglia può essere pensato come un iperpiano che divide lo spazio in due parti o sottospazi. Considerando tali nodi, i punti a sinistra dell'iperpiano rappresentano il sotto albero di sinistra, e punti di destra dell'iperpiano rappresentano il sotto albero di destra. La direzione iperpiano viene scelta nel modo seguente: ogni nodo dell'albero è associato con una delle k-dimensioni, con l'iperpiano perpendicolare all'asse che la dimensione. Così, per esempio, se si sceglie un punto di divisione sull'asse x, tutti i punti della sottostruttura con x di valore più piccolo rispetto al nodo, appariranno nel sotto albero sinistro e tutti i punti con i valori più grandi di x verranno posti nel sotto albero di destra. In tal caso, l'iperpiano sarebbe fissato dal punto x, e la sua normale sarebbe l'unità di ascissa x.

5.6Come costruire la Probabilistic Roadmap

La costruzione del kd-tree nell'algoritmo k-nearest neighbors avviene nel seguente modo :

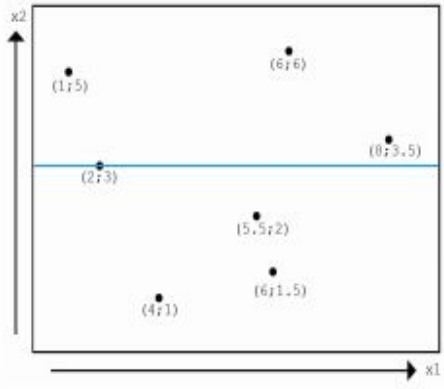
1. in nodo radice è il punto di query, e dato questo punto di divide lo spazio in due iperpiani ;
2. si seleziona un punto, in base alla posizione che possiede viene inserito nel sottoalbero di destra o sinistra, l'inserimento viene fatto in maniera bilanciata, con riferimento alla distanza euclidea ;
3. si itera il punto 2, finché non sono stati analizzati tutti i punti.

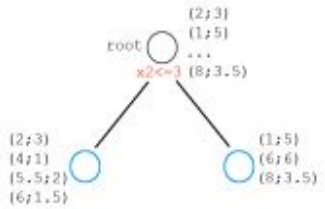
Qui di seguito è illustrato il procedimento precedentemente descritto.



PASSO 1

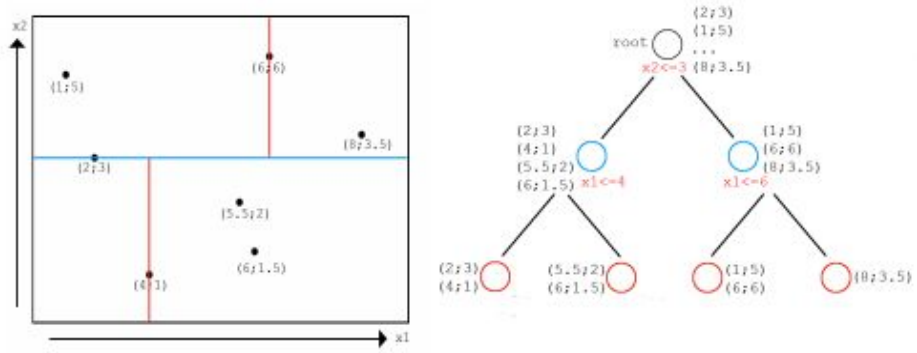
Si definiscono i punti sullo spazio.





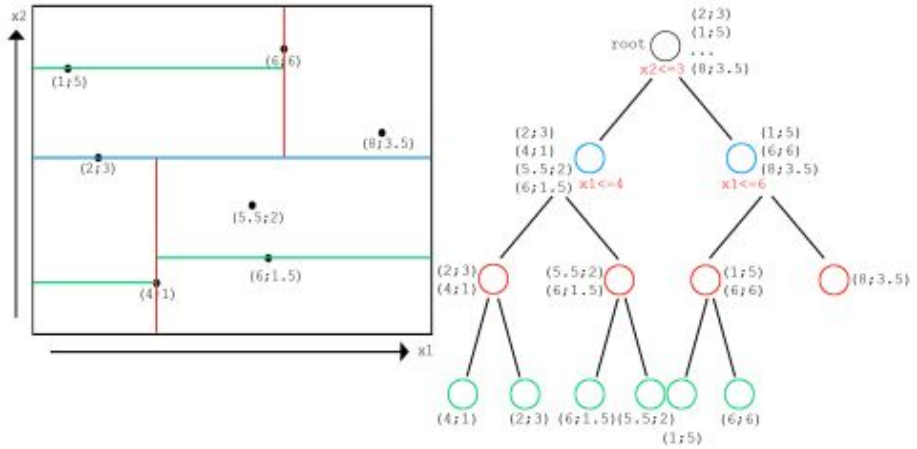
PASSO 2

Selezionato il punto di partenza, si traccia il primo semipiano divisore e si posizionano i punti sul sottoalbero di destra o di sinistra in base alla distanza euclidea.



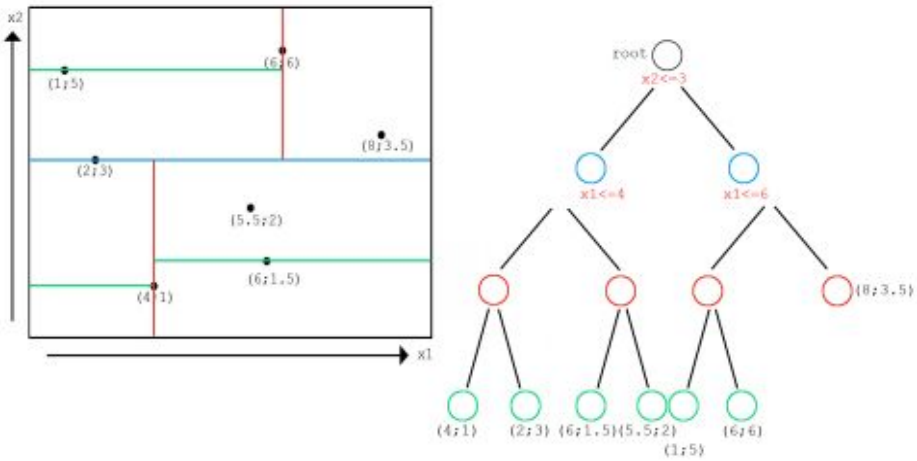
PASSO 3

Si iterano le operazione specificate nel passo 2.



PASSO 4

Si iterano le operazione specificate nel passo 2.



PASSO 5

Alla fine del processo i punti sono memorizzati solo sulle foglie, e per sapere dove si trova un punto occorre scorrere tutto il sottoalbero dove è situato.

Figura 5.10 Funzionamento kd-tree

Capitolo 6

Algoritmo genetico applicato al problema del protein folding

I risultati dell'applicazione dell'algoritmo genetico classico sul problema di predizione della struttura proteica, non sono i migliori[7,12, 32]. Tuttavia gli algoritmi genetici sono considerati strumenti interessanti per il problema del protein folding. Questo perché con la sola integrazione all'interno dell'algoritmo genetico di metodi cooperativi, la qualità delle soluzioni dell'algoritmo genetico possono notevolmente migliorare. Per esempio, considerando sempre il problema della predizione proteica, lo studio della configurazioni spaziali, lo studio delle configurazioni energetiche e la reimplementazione degli operatori genetici, possono essere modificate per essere finalizzate ad una ricerca dell'ottimo per la struttura proteica, anziché per le stringhe come previsto della teoria dell'algoritmo genetico. Queste nuove implementazioni dell'algoritmo genetico ha portato alla formulazione di numerosi algoritmi genetici differenti [10, 13, 16, 33, 34]. Solitamente questi metodi cooperativi agiscono nella “lacuna” maggiore dell'algoritmo genetico, ovvero il fatto l'algoritmo genetico arriva a molti risultati sono buoni ma non ottimi. Per ovviare a ciò, si applicano operazioni di ottimizzazioni locali della struttura proteica.

In questo lavoro, oltre a definire un operatore genetico (la mutazione) per la ricerca locale, si applica anche un metodo cooperativo per lo studio energetico delle configurazioni che compongono la popolazione, nell'energy landscape.

Verrà esposto ora, come è stato formalizzato il problema e come è stato implementato l'algoritmo.

6.1 Formalismo

Il cromosoma rappresenta le coordinate cartesiane tridimensionali di una configurazione della proteina sul reticolo. Ogni configurazione è definita nello spazio discreto (reticolo) che

come è stato definito il precedenza è lo *spazio delle configurazioni*.

In accordo col lavoro di Krasnogor [14] il cromosoma identifica la struttura tridimensionale della proteina attraverso coordinate cartesiane relative. Il cromosoma, infatti, è una stringa definita sull'alfabeto A := { U, D, L, R, B, F }, con la seguente legenda :

- U = Up (in alto)
- D = Down (in basso)
- L = Left (a sinistra)
- R = Right (a destra)
- B = Back (indietro)
- F = Forward (avanti)

Questa codifica ha il vantaggio di convertire facilmente convertiti strutture tridimensionali in stringa o viceversa. Tuttavia, essa ha lo svantaggio che gli operatori genetici in molti casi possono creare conformazioni della proteine non valide, in quanto vi possono essere delle collisioni. Pertanto è necessario progettare un filtro che elimina gli individui non validi. Di contro avremo un deteriorarsi delle prestazioni dell'algoritmo se il numero di conformazioni che collidono sono numerose.

Per considerare veri tutti i fondamenti matematici su cui si basa l'algoritmo genetico, deve essere possibile effettuare una mappatura del cromosoma ad una stringa binaria; questo perché la teoria dell'algoritmo genetico è dimostrata per rappresentazioni binarie.

6.2 Funzione di Fitness

La funzione di fitness è quella definita nel capitolo 3, descritta nel lavoro di Berenboym e Avigal [16].

La scelta di questa funzione, preferita alla funzione di fitness HP standard, è motivata dall'esigenza di poter discriminare meglio le conformazioni. Per discriminazione si intende avere valori di fitness differenti con strutture del nucleo idrofobico diverse. Per esempio, date due configurazioni c_1 e c_2, esse possono aver stesso valore di funzione di fitness ma avere configurazioni differente, come indicato in figura. Ciò è dovuto del fatto che le informazioni della struttura del nucleo idrofobico vengono date solamente dai contatti H-H, mentre

informazioni discriminanti per strutture che possono formare gli amminoacidi polari non se ne possono ricavare.

(a) (b)

Figura 6.1 Conformazioni 3D

La conformazione (a) e la conformazione (b) hanno stesso valore di energia idrofobica se il calcolo è eseguito secondo lo standard per il modello HP, quindi è impossibile discriminare le due conformazioni. Tuttavia osservando le due conformazioni si nota chiaramente che la conformazione (b) è "migliore" in quanto presenta una struttura nel nucleo idrofobico più definita.
Se si consedera l'energia globale, ovvero il modello energetico Berenboym e Avigal, si osserva che la conformazione (a) ha energia globale minore alla conformazione (b). Questo tipo di modellazione favorisce la convergenza a conformazioni con nucleo idrofobico maggiormente definito, ed è per questo motivo che è stata utilizzata in questo lavoro di tesi.

Date questa esigenza, considerare anche i contatti H-H diagonali, è utile per discriminare maggiormente le configurazioni spaziali del nucleo idrofobico. I casi di valori di fitness coincidenti per conformazioni differenti diminuiscono in maniera significativa con questa definizione di funzione di fitness.

6.3 Operatori Genetici

Gli operatori definiti in questo lavoro, sono un single point crossover ed un operatore mutazione descritto dall'insieme delle mosse *Pull Move*.

6.3.1 Single Point Crossover

Il crossover single point spezza due geni e li riunisce generando due figli. In questo caso, il crossover viene definito sul reticolo cubico, ed a differenza del caso teorico si deve gestire eventuali collisioni delle conformazioni dei figli. Se si genera delle collisioni, si ripete l'operazione di crossover, finché non va a buon fine o finché non si arriva ad un numero

massimo di tentativi.

6.3.2 Mutazione

L'operatore mutazione, dalla descrizione che viene data dal punto di vista teorico, dovrebbe permettere di sfuggire dai minimi locali, ma in questo caso la mutazione serve per arrivare a dei minimi. Per fare ciò, si adotta una tecnica di ricerca locale, l'insieme di mosse Pull Move.

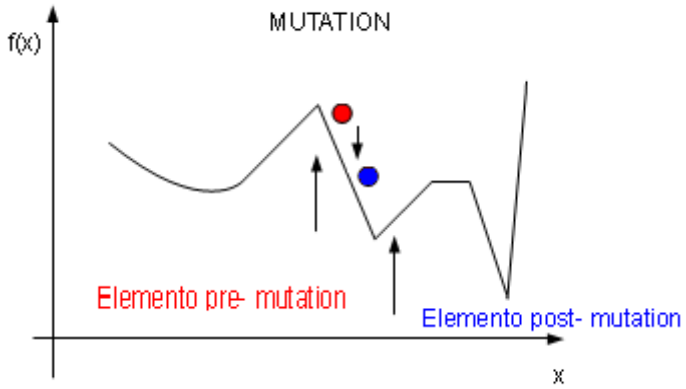


Figura 6.2 La mossa Pull Move

L'effetto della mossa Pull Move è quello di esplorare in maniera migliore lo spazio locale, per giungere ai minimi locali della funzione energia.

Gli algoritmi standard soffrono di una incapacità di avere delle modifiche locali efficienti sulla struttura, al fine di migliorare l'energia complessiva del posizionamento degli amminoacidi. Questo succede perché gli operatori crossover e mutazione, nella definizione standard, non sono abbastanza “flessibili”. Si è pensato dunque di applicare una insieme di Pull Moves prima di ogni riproduzione della popolazione, e si è considerato questa operazione come un'operazione di mutazione.

Ogni amminoacidi può essere soggetto ad una Pull Move. Intuitivamente, l'operazione di “pull” diagonale di un amminoacido, causa una serie di altri movimenti sulla catena amminoacidica di modo che la struttura risulti ancora valida.

Nella maggior parte dei casi, la quantità di amminoacidi spostati è piccola. Così da ritenere senza dubbio che le mosse Pull Move siano da considerare con modifiche locali.

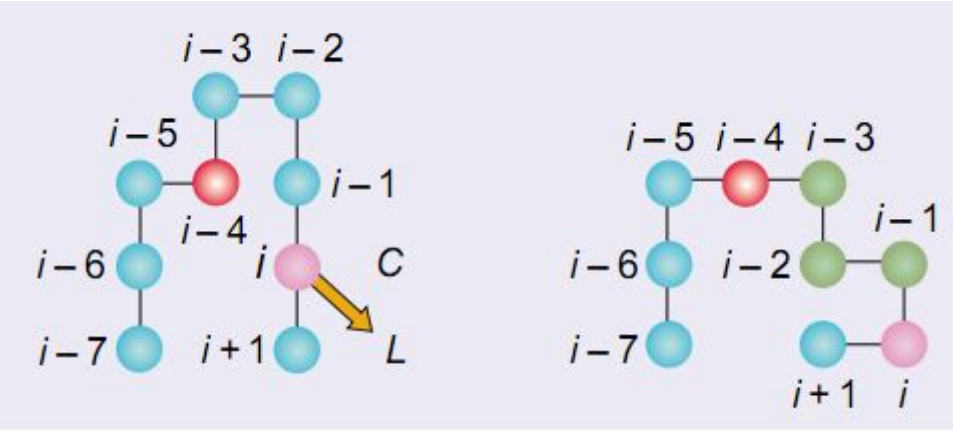


Figura 6.3 Esempio Pull Move diagonale

6.3.3 Pull Move

L'insieme delle mosse Pull Move viene descritto da Lesh [35] ed è un insieme completo, reversibile e locale.

Per prima cosa definiamo cosa si intende con il termine mossa. Data una conformazione *c* valida della struttura della proteina definita sul reticolo, si definisce mossa un cambiamento della struttura che comporta ad una nuova configurazione *c′* valida. Le caratteristiche dell'insieme Pull Move sono :

- *reversibile*, se esiste una mossa deve esistere anche la sua inversa;
- *completo*, se data una qualsiasi coppia di configurazioni *c* e *c′*, esiste una sequenza di mosse in M che trasforma *c* in *c′*;
- *locale*, data una posizione *i* nella configurazione queste mosse muovono il minor numero possibile di amminoacidi e in posizioni non troppo distanti dalla posizione iniziale *i*;

Definizione Formale in uno spazio bidimensionale:

Si definisce una locazione come un nodo del reticolo corrispondente alla coordinate (x, y). Due locazioni si dicono adiacenti se sono tra loro divise da un singolo lato verticale o orizzontale, e diagonalmente adiacenti se sono tra loro divise da una lato orizzontale e poi da uno verticale, o viceversa.

La sequenza di palline, che schematizzino la sequenza amminoacidica, è numerata da 1 a N e con la notazione $(x_i(t), y_i(t))$ si indica che la locazione (x, y) al tempo t è occupata dall'i-

esimo amminoacido.

Una configurazione valida della catena al tempo t si indica con il simbolo c(t). Inoltre, il sottopercorso di c(t) dall'amminoacido i all'amminoacido j inclusi, con j > i, si indica con $c_{i,j}(t)$.

Supponiamo che una locazione libera L sia adiacente a $(x_{i+1}(t), y_{i+1}(t))$ e diagonalmente adiacente a $(x_i(t), y_i(t))$. Le locazioni $(x_i(t), y_i(t))$, $(x_{i+1}(t), y_{i+1}(t))$ e L formano allora tre angoli di un quadrato, sia il quarto angolo la locazione C. Anche si possa eseguire una Pull Move la locazione C deve essere libera o equivalente a $(x_{i-1}(t), y_{i-1}(t))$.

Quando $C = (x_{i-1}(t), y_{i-1}(t))$, l'intera Pull Move consiste nel muovere l'amminoacido i nella locazione L. Quando C è libera, i viene mosso nella locazione L e $i - 1$ viene mosso nella locazione C. Poi, fino a quando non si raggiunge una configurazione valida si esegue la seguente operazione, iniziando dall'amminoacido $j = i - 2$, fino all'amminoacido 1 si pone $(x_j(t + 1), y_j(t + 1)) = (x_{j+2}(t), y_{j+2}(t))$.

Gli aminoacidi sono cioè mossi di due posizioni in su nella catena fino a raggiungere una configurazione valida. Si noti che in questo modo si ha la sicurezza di mantenere una configurazione valida, infatti gli aminoacidi i e $i - 1$ sono mossi in locazioni libere mentre quelli di indice minore sono ripetutamente mossi in locazioni che vengono di volta in volta sgombrati.

La Pull Move termina appena si ottiene una configurazione valida, nel peggiore dei casi si scorre tutta la catena fino ad arrivare all'amminoacido 1, a quel punto la configurazione ottenuta sarà sicuramente valida. Il motivo per cui ci si ferma appena si ottiene una configurazione valida non è tanto per motivi di prestazioni computazionali ma perché muovendo meno aminoacidi si ha un miglioramento della località della mossa.

Abbiamo descritto una Pull Move dall'amminoacido i giù fino all'amminoacido 1, allo stesso modo ci si può muovere nell'altra direzione, cioè verso N, iniziando da una locazione L adiacente a $(x_{i-1}(t), y_{i-1}(t))$ e diagonalmente adiacente a $(x_i(t), y_i(t))$

Infine, per rendere l'insieme di Pull Move reversibile, bisogna aggiungere delle altre mosse speciali per gli aminoacidi finali della catena. Si considerino due locazioni libere tra loro adiacenti e inoltre una di queste locazioni sia adiacente all'ultimo amminoacido N. Possiamo allora muovere gli aminoacidi $N - 1$ e N in queste locazioni, e in seguito trascinare gli aminoacidi rimanenti fino a ottenere una configurazione valida: iniziando dall'amminoacido $j = N - 2$ e giù fino all'amminoacido 1 si pone $(x_j(t + 1), y_j(t + 1)) = (x_{j+2}(t),$

$y_{j+2}(t)$). In modo simile si possono eseguire queste mosse nella direzione opposta, a partire cioè dall'amminoacido 1.

6.4 Selezione

Il processo di selezione, rappresenta l'elemento di interesse e di novità del lavoro di tesi. In questa fase, si passa dallo spazio delle conformazioni allo spazio energetico, per poter studiare l'energy landscape delle configurazioni e poter selezionare gli individui con un nuovo criterio.

Lo studio dello spazio energetico deve fornire ulteriori informazioni riguardo la scelta di individui adatti per la riproduzione della popolazione (crossover e mutazione). Per fare ciò, anziché considerare solo il valore di fitness, verranno studiate il posizionamento di tutti gli individui sull'energy landscape in modo da scegliere quelli che sembrano più promettenti per una convergenza verso vari punti di minimo. L'energy landscape, come già detto nel capitolo 5, definisce la superficie energetica di tutte le configurazioni valide e possibili data una sequenza di amminoacidi.

Le considerazioni che si possono effettuare studiando l'energy landscape, sono che un buon individuo per la riproduzione è un individuo che possiede un posizionamento sulla superficie energetica vicino ad un minimo e che i punti selezionati non siano energeticamente vicini.

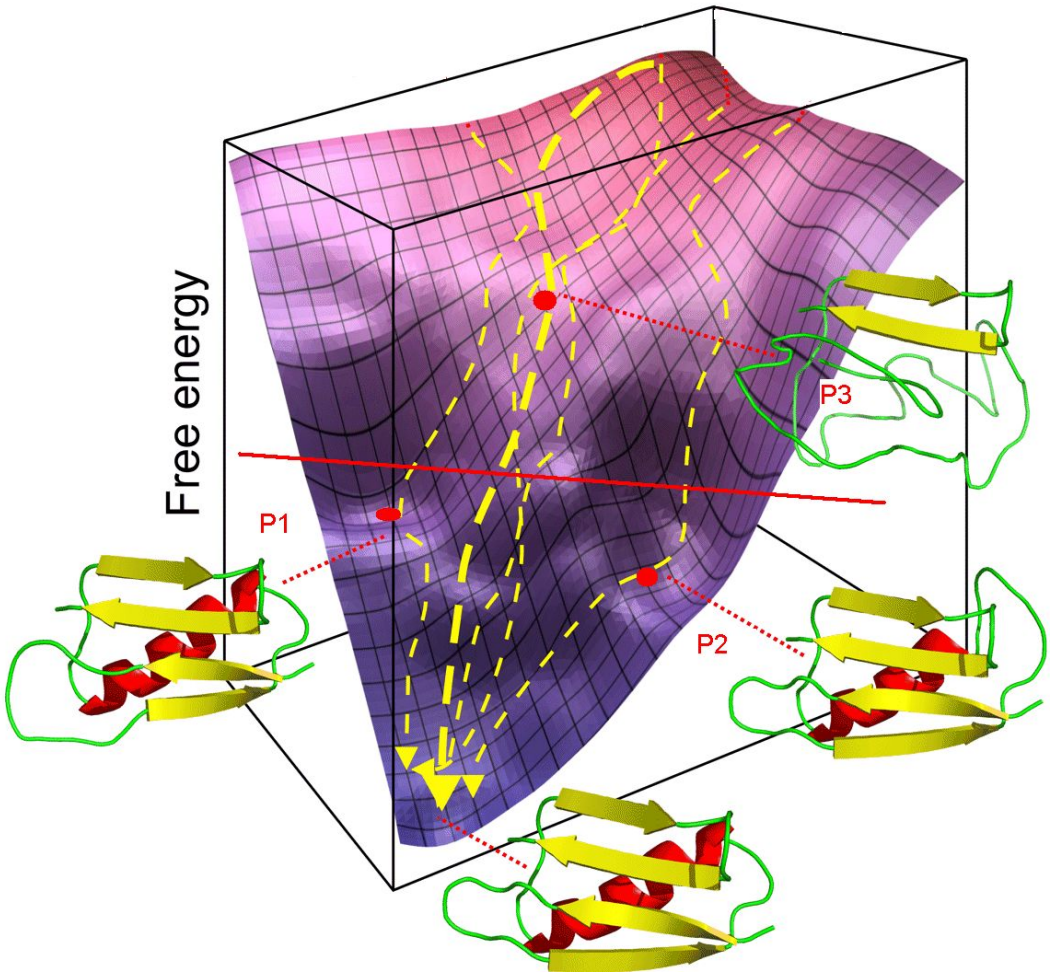


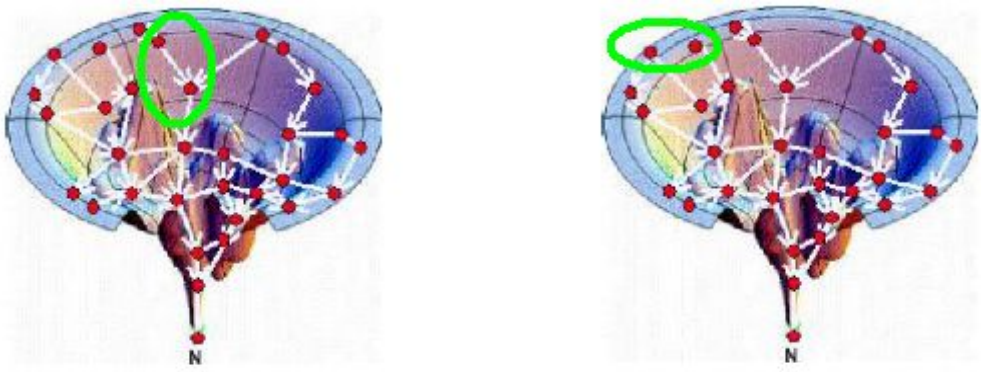
Figura 6.4 Scelta errata dei punti nell'energy landscape

Nella figura sopra si evidenzia il problema della selezione errata di punti per la riproduzione della popolazione. I punti migliori solo quelli che minimizzano la funzione di fitness, è facile accorgesi che il punto P3 non viene selezionato, in quanto a valore di fitness elevato rispetto gli altri. Però così facendo si elimina dalla popolazione una conformazione che può convergere all'ottimo in tempi rapida, così come mostrato in figura.
La linea rossa orizzontale indica il livello della funzione di fitness di selezione, mentre le linee tratteggiate in giallo indicano i percorsi ottimali del ripiegamento.

La condizione di essere vicino ad un punto di minimo, è soddisfatta considerando solamente un punto con valore di fitness minimo. Questo perché la coordinata di profondità nella rappresentazione dell'energy landscape è descritta dall'energia idrofobica.

Per quanto riguarda la vicinanza energetica, si deve richiamare il concetto di transizione energetica e la sua fattibilità, esposta nel capitolo 5. Due punti p_1 e p_2 possono essere topologicamente vicini sulla superficie energetica, si possono presentare due opzioni come

mostrato in figura 6.5.



(a) (b)

Figura 6.5 Collegamento nella Probabilistic Roadmap

Nella figura 6.5 (b) sono segnalati i punti p_1 e p_2 che non sono collegati nell'interno della stessa roadmap, mentre nella figura 6.5 (a) sono segnalati punti con la stessa distanza euclidea del punto b ma che sono collegati nella stessa roadmap. Ipotizzato che i punti in questione siano i punti con minor valore di fitness di tutta la popolazione con $p_1 < p_2$, nel primo caso i due punti vengono considerarti per la fase di riproduzione, mentre nel secondo caso, viene considerando solo p_1. Questo perché tra p_1 e p_2 esiste una transizione energetica fattibile e ciò implica che a livello di informazioni i cromosoma relativi a p_1 e p_2 contengono le stesse informazioni, quindi si può salvare solamente il patrimonio genetico nel migliore dei due, senza correre il rischio di perdere informazioni sulla popolazione.

Il metodo pensato funziona nella seguente maniera:

1. *si seleziona la conformazione a minor energia idrofobica;*
2. *si crea la probabilistic roadmap ad essa associata;*
3. *si mantengono le connessione della probabilistic roadmap, che hanno fattibilità energetica (probabilità) maggiore del 50%;*
4. *si elimina dalla lista delle conformazione da studiare tutte quelle connesse alla probabilistic roadmap del punto precedente;*
5. *si esegue il passo 1 con le conformazione rimaste, finché non siano tutte esaminate;*

Al termine di questo procedimento avremo l'energy landscape suddiviso in diversi grafi o punti isolati. Per ogni suddivisione si seleziona la conformazione con energia idrofobica minore, oltre ai punti singoli, e questi faranno parte della popolazione ritenuta maggiormente adeguata per il crossover e la mutazione.

Capitolo 7

Struttura del software

7.1 GAlib

GAlib [36] è una libreria di oggetti di algoritmi genetici, che può essere usato nei programmi C++ usando qualsiasi rappresentazione e qualsiasi operatore genetico. Si può trovarla disponibile per il download all'indirizzo internet : http://lancet.mit.edu/ga , è stata implementata da Matthew Wall del Mechanical Engineering Department del Massachusetts Institute of Technology.

Questa libreria (al cui interno vi sono classi template) è disponibile in codice sorgente, permette così di essere usata in vari sistemi (Windows, MacOS e Unix). Essa consiste in un insieme di classi base, che inglobano tutti gli elementi più importanti di un algoritmo genetico tradizionale, e numerose altre classi specifiche, derivate dalle stesse classi basi.

<table>
<tr><td>GAGeneticAlgoritm
GASimpleGA
GASteadyStateGA
GAIncrementalGA
GADemeGA</td><td>GASelectionScheme
GARankSelector
GARouletteWheelSelector
GATournamentSelector
GAUniformSelector
GASRSSelector
GADSSelector</td><td>GAScalingScheme
GANOScaling
GALinearScaling
GAPowerLawScaling
GASharing
GASigmaTruncationScaling</td></tr>
<tr><td rowspan="2">GAStatistics
GAParameterList
GAPopulation

GAList<T>
GAListGenome<T>

GATree<T>
GATreeGenome<T></td><td colspan="2">GAGenome Derived Classes:</td></tr>
<tr><td>GABinaryString
GA1DbinaryStringGenome
GABin2DecGenome
GA2DbinaryStringGenome
GA3DbinaryStringGenome</td><td>GAArray<T>
GA1DArrayGenome<T>
GA1DArrayAlleleGenome<T>
GA2DArrayGenome<T>
GA2DArrayAlleleGenome<T>
GA3DArrayGenome<T></td></tr>
</table>

Figura 7.1 Gerarchia delle classi della libreria GAlib

La libreria GAlib presenta le seguenti caratteristiche :

- Può essere usata per evolvere popolazione e/o individui in parallelo, impiegando CPU multiple con l'uso di *"Parallel Virtual Machine" (PVM)*.
- Molte variazione congenite degli algoritmi genetici sono fornite comprese tipi

popolazione, terminazione, strategie di rimpiazzo, meccanismi di selezione, inizializzazione dei cromosomi e operatori genetici, tutti questi possono essere personalizzati o rimpiazzati.

- La registrazione delle varie statistiche è disponibile e include misurazioni on-line e off-line ed i loro minimi e massimi.

Le classi GA contengono statistiche, la strategia di rimpiazzo e parametri operazionali. Un oggetto popolazione è usato come contenitore della classe genoma, della classe statistiche (relative alla popolazione), della classe strategia di selezione e della classe della scaling strategy. L'utente deve almeno fornire la funzione obiettivo e la funzione di inizializzazione delle classi e come eseguire la ricerca.

7.1.1 Le classi principali

GAGenome

La classe *genome* è usata per incapsulare il genoma e le funzioni che operano sul genoma. Le funzioni di statistica sono usate per definire l'insieme di base dei metodi della classe e include i metodi di inizializzazione, di comparazione, di mutazione, di crossover e di valutazione. Questi sono metodi di default dato che sono stata progettati per essere ignorati dalla funzioni alternative in run-time. Altri metodi addizionali sono per copiare e clonare i genomi, per caricare e salvare i dati in un stream, e , opzionalmente, una funzione comparatore può essere definita per calcolare il grado di differenza tra genomi. La classe genome in genere dovrebbe utilizzare l'ereditarietà multipla alle sottoclassi della classe base genome (GAGenome) e alle classe data type (ad esempio, GABinaryString o altre classi definite dall'utente).

GAPopulation

Questa classe agisce come un contenitore per i genomi. Dispone di metodi per l'inizializzazione, valutazione, calcolo statistiche, selezione e scaling.

GAGeneticAlgorithm

Classi alternative GA possono essere create come sottoclassi da una qualsiasi delle classi esistenti, modificandole riscrivendo uno o più dei loro metodi.

Derivazioni di queste classi sono usate per determinare come le generazioni della popolazione sono costruite, per esempio con steady steate o con metodo incrementale. Questa classe anche contiene funzioni di terminazione statiche, che hanno una classe GA passata a loro perciò possono richiedere alcuni dati rilevanti, e devono ritornare o vero o falso per indicare se il GA dovrebbe terminare.

GAScalingScheme

Questa classe trasforma la funzione obiettivo "grezza" in valori di scaled fitness. Le classi scaling possono essere create come sottoclassi della classe astratta GAScalingScheme. Una funzione di valutazione è usata per calcolare il valore di scaling sul valore originale di fitness. Per tipi di dato non banali, le funzioni clone e copy devono essere implementate.

GASelectionScheme

Questa classe è usata per calcolare la funzione che estrae i genomi dalla popolazione, che saranno usati per la riproduzione della popolazione stessa. Altri schemi di selezione possono essere creati derivando una delle classi di selezione già esistenti. Ci sono due metodi da definire : update e select. Il metodo update è chiamato prima della selezione in modo che tutta la fase di pre-selezione sia eseguita correttamente. Il metodo select estrae un individuo dalla popolazione , ritornando un riferimento al genoma.

In questo modo la libreria stabilisce i mezzi per usare e modificare un GA, ed include molte delle varianti tradizionali per il GA oltre a numerosi tipi dato differenti per il genoma.

7.2 BIU - Bioinformatic Utility Library

BIU [37] è una libreria C + + che racchiude le classi e le funzioni, che molto spesso sono usate in contesto bioinformatico. Raccoglie le strutture dati e le gerarchie di classi per la modellazione di biomolecole e funzionalità generali utili, come il parsing dei parametri del programma, le conversioni di stringa, ecc... Implementata da Martin Mann, Sebastian Will, Daniel Maticzka e Hannes Kochiß, dell' Università di Friburgo, è disponibile in codice sorgente all'indirizzo web : http://www.bioinf.uni-freiburg.de/Software/Libraries/index.html?en#lib_biu .

7.2.1 Funzionalità

Operazioni su stringhe

Lo standard C++ ha predefinite solo poche operazioni su stringhe con chiamata immediata. In questa libreria sono definite un insieme di funzioni wrapper basate sulle funzionalità STL per un uso semplice.

Parsing dei parametri del programma

Gestione ed analisi dei parametri di parsing.

Generazione di numeri casuali

La generazioni di numeri pseudo-casuali è una parte importate per la simulazione di situazioni biologiche/chimiche. Sono disponibili le seguenti tipologie di generazione random :

- una funzione wrapper su ISO C (risultati CPU dipendente) ;
- generatore lineare congruente ;
- una funzione wrapper derivata da GNU Scientific Library .

Reticolo Proteico

La libreria dispone di rappresentazioni su reticolo per le proteina con funzioni energetiche basata sui contatti, in reticoli da scegliere. Le possibilità di scelta dei reticoli sono i seguenti :

- 2D, reticolo quadro
- 3D, reticolo cubico
- 3D, reticolo cubico a facce centrate (FCC)

La funzione energetica può essere definita su un arbitrario alfabeto. Si supportano differente funzioni energetica basate sulla distanza :

- contact based (solo le posizioni vicine nel reticolo danno un contributo)
- distance interval based (differente contributo energetico per intervalli di distanza discreti)

Per mosse della struttura nel reticolo sono implementate le seguenti:

- pull moves ;
- pivot moves .

7.2.2 Le classi

biu::AllowedBasePairs biu::Alphabet biu::Alphabet::hash_string biu::BackboneStructure3D biu::LatticeProtein biu::LatticeProtein_I biu::LatticeProtein_Ipnt biu::OffLatticeProtein biu::BioMolecule biu::LatticeProtein biu::RNAStructure biu::RNAStructure_TB biu::COption biu::COptionParser biu::COptionParser::hash_string biu::util::CPrintable biu::util::CException biu::DistanceEnergyFunction biu::ContactEnergyFunction biu::IntervalEnergyFunction	biu::Graph_UD gsl_qrng gsl_qrng_type biu::LatticeDescriptor biu::LatticeDescriptorCUB biu::LatticeDescriptorFCC biu::LatticeDescriptorSQR biu::LatticeDescriptor::AutomorphismData biu::LatticeDescriptor::NeighborData biu::LatticeModel biu::LatticeFrame biu::LatticeMoveSet biu::PivotMoveSet biu::PullMoveSet biu::LatticeNeighborhood biu::LatticeNeighborhoodCUB biu::LatticeNeighborhood::hash_IntPoint biu::Matrix< T > biu::SquareMatrix< T, matrixDim > biu::Point3D< T >

biu::RandomNumberGenerator biu::RNG_ISO biu::RNG_LCG biu::RNAStructure_TB::TreeItem biu::Timer biu::util::Util_String biu::VirtualList< T > biu::VirtualList< T >::Iterator biu::VirtualList< T >::ItState	biu::Point3D< int > biu::NeighborVector biu::PullMoveSet::PullMoveDecoder biu::PullMoveSet::RelativeInt biu::PullMoveSet::RelativeInt::Operators biu::PullMoveSet::RelativeInt::InvOps biu::PullMoveSet::RelativeInt::StdOps biu::PullMoveSet::UndoRecord biu::QuasiRandomNumberGenerator biu::RandomNumberFactory

Figura7.2 Gerarchia classi

7.3 ANN- Library for Approximate Nearest Neighbor Searching

ANN è una libreria scritta in C + +, che supporta le strutture dati e algoritmi per ricerche nearest neighbor esatte e approssimative in dimensioni arbitrariamente elevate. Implementata da David M. Mount dell'University of Maryland e Sunil Arya di Hong Kong University of Science and Technology. La libreria ANN si trova disponibile per il download all'indirizzo web : http://www.cs.umd.edu/~mount/ANN/ .
Come input del problema deve essere dato un insieme di punti in uno spazio *d*-dimensionale. Questi punti sono pre-elaborati in una struttura di dati, in modo che dato qualsiasi punto query *q*, i *k* punti più vicini possano essere trovati in maniere rapida. La distanza tra due punti può essere definita in molti modi. ANN presuppone che le distanze sono misurate con una classe di funzioni distanza denominata metriche di Minkowski. Questi includono anche le distanze note, come la euclidea, la distanza di Manhattan, e la distanza max.
La libreria implementa una serie di diverse strutture di dati, basate su kd-alberi e decomposizione in celle, ed implementa un paio di strategie di ricerca diverse.

L'elemento basico da manipolare con ANN è il punto, che viene definito come un array non dimensionato, o un puntatore alle coordinate.

typedef ANNcoord ANNpoint; // a point*

Altre strutture dati definite sono l'array di punti e l'array di distanza (float), anche questi sono definiti come array non dimensionato.

typedef ANNpoint ANNpointArray; // an array of points*

typedef ANNdist ANNdistArray; // an array of squared distances*

Infine è definito anche l'array di indice, che è un array di interi non dimensionato.

typedef ANNidx ANNidxArray; // an array of point indices*

Infine ANN dispone anche di un tipo dato boolenao, ANNbool, con i valori ANNfalse e ANNtrue.

7.3.1 Nearest Neighbor Search Structure

ANN dispone di tre differenti strutture di ricerca su insiemi ANNpointSet:

ANNbruteForce, ANNkd_tree,e ANNbd_tree.

La struttura di ricerca che verrà utilizzata è ANNkd_tree. Questa struttura supporta le seguenti operazioni.

- *Constructor*: Costruisce un kd-tree da un insieme di n punti in un spazio d-dimensionale.

```
ANNkd_tree::ANNkd_tree(
        ANNpointArray pa,      // data point array
        int n,   // number of points
        int d);   // dimension
```

- *k-Nearest Neighbor Search*: Ann implementa due tipi differenti di ricerca sull'albero, normale e prioritaria. Questa funzione dato un insieme di punti query, il numero di vicini da trovare, e l'errore di bound; restituisce due array, il primo contenente la lista dei punti vicini, e il secondo contenete la distanza di questi punti.

```
virtual void ANNkd_tree::annkSearch(
        ANNpoint q, // query point
        int k, // number of near neighbors to find
        ANNidxArray nn_idx, // nearest neighbor array (modified)
        ANNdistArray dists, // dist to near neighbors (modified)
        double eps=0.0); // error bound
```

- *Fixed-radius k-Nearest Neighbor Search*: implementa la stessa della funzione descritta precedemente, con la differenza che la ricerca dei vicini si effettua all'interno di uno spazio fisso.

7.4 Implementazione

Per prima cosa si è definito il genoma da utilizzare nella simulazione. Si è quindi derivata la classe MyGenome dalla classe GAGenome definita nella libreria GAlib.

Il genoma, rappresenta la struttura della proteina sul reticolo, per questo nella sua definizione vengono usati gli oggetti definiti nella libreria BIU. Inoltre, sono state aggiunte due funzioni, una che calcola l'entropia della conformazione, e l'altra che calcola l'energia libera.

Per quanto riguarda la fase di selezione, è stata derivata dalla classe GAScaling, una nuova classe MyScaling, dove è implementato il metodo Roadmaps. Nella definizione della classe MyScaling, è stata utilizzata la libreria ANN, nella fase di esplorazione dello spazio energetico.

Infine, nel file main, si crea l'oggetto algoritmo genetico, specificando i parametri e le funzioni da utilizzare.

7.4.1 MyGenome

La classe MyGenome è derivata dalla classe GAGenome, presente nella libreria GAlib. Per poter definire questa classe occorre reimplementare alcuni metodi definiti in GAGenome. Nello specifico, in questo lavoro, sono stati ridefiniti i seguenti metodi : initializer, mutator, evaluate, crossover, MyGenome, ~MyGenome,clone e copy; oltre alla definizione di altri due metodi specifici per questa tesi : entropy e freEnergy.

MyGenome è il costruttore dell'oggetto genoma, ~MyGenome è il distruttore, copy e clone si riferisco alla copie e clonazione di un oggetto genoma già esistente. Tutte questi metodi sono stati ridefiniti nella maniera classica, così come descritto nel manuale di GAlib, in quanto solo metodi che lavorano solamente con il puntatore all'oggetto genome e non con la struttura dell'oggetto.

Il metodo initializer, implementa l'inizializzazione dell'oggetto genome, la quale consiste nella creazione dell'oggetto reticolo e nell'assegnazione di una conformazione random della proteina sul reticolo stesso. Per fare ciò si utilizzano gli oggetti come LatticeDescriptorCUB resi disponibili dalla libreria BIU, ai quali si deve specificare che tipo

di alfabeto per le sequenze di amminoacidi viene usato, la matrice energia, ecc. L'uso della libreria BIU rende più semplice il test se una conformazione creata sia valida o meno, in quanto vi è implementato il metodo *isValid()*. La creazione dell'oggetto genome è completata se a sua volta la creazione della conformazione proteica è valida, il processo di creazione è dunque il seguente:

1. si crea l'oggetto definito da BIU che definisce la configurazione;
2. si crea una stringa di mosse[10] in modo random;
3. si testa che la configurazione creata seguendo la sequenza di mosse sia valida;
4. se non è valida la configurazione si ripete dal punto 2.

Il metodo crossover reimplementato è, come specificato nel par. 6.3.1, il single crossover. Si effettua la combinazione delle due stringhe contenenti la sequenza di mosse dei genitori, quindi un normale crossover su stringhe; ma successivamente si deve testare che le due sequenze di mosse figlie descrivano configurazioni proteiche valide. Nel caso nessuno dei due figli non sia valido, si ripete il crossover prendendo un altro punto di break sulle sequenze genitori, sempre in maniera random, e si ripete la verifica della validità.

La reimplentazione del metodo mutator consiste solamente del chiamare il metodo pull move presente nella libreria BIU, esso realizza la mossa e i controlli di conformazione valida.

Per quanto riguarda il metodo evaluate, rientra nell'ambito dei "metodo energetici", assieme al metodo freRnergy e entropy. Questi metodo implementano il calcolo di energia idrofobica, di energia libreria e di entropia che possiede la conformazione, per fare ciò occorre definire una nuova struttura dati all'interno della classe MyGenome. La struttura dati definita è un array a 3 dimensioni di caratteri, avendo così due strutture dati che definiscono la struttura proteica. Questo è necessario perché la libreria BIU non permette modifiche al proprio codice in maniera modulare come la libreria GAlib, e risulta più semplice definire il calcolo di questi valori su una nuova struttura dati. La funzione evaluate viene calcolata con la struttura dati array anziché richiamando la funzione prevista dalla libreria BIU; perché la libreria BIU non permette la definizione del modello energetico di Berenboym e Avigal [6], ma solamente la funzione energetica classica, senza contatti diagonali.

10 Con il termine **stringa (o sequenza) di mosse** si intende una successione di caratteri definiti sul seguente alfabeto {U, D, L, R, F, B}, che identificano i movimenti della sequenza di amminoacidi in coordiante relative.

7.4.2 MyScaling

La classe MyScaling è derivata dalla classe GAScalingScheme, presente nella libreria Galib. Questa classe implementa la classificazione della popolazione per la successiva fase di crossover e mutazione. La classificazione implementata è quella descritta nel capitolo 6, e per fare ciò occorre utilizzare alcune funzione implementate nella libreria ANN.

Si sono reimplementate le funzioni evaluate e clone. La funzione clone semplicemente richiama crea un nuovo oggetto MyScaling. La parte più interessante di questa classe è indubbiamente la realizzazione della funzione evaluate.

Per prima cosa si prepara un array bidimensionale che contengono i valori di entalpia e di energia libera di ogni configurazione della popolazione, da passare successivamente alle funzioni definite da ANN. In pratica questo array definisce i punti nello spazio bidimensionale energetico (energy landscape) che ha come coordinate l'energia libera e l'entropia. Creato il kd-tree e tutte le strutture per la ricerca si creano le probabilistic roadmaps. Si parte dai punti a minor valore di energia idrofobica, si trovano i vicini nell'energy landscape e se il collegamento è energeticamente fattibile restano aggregati alla probabilist roadmap. Il valore di ranking sulla popolazione viene settando nelle prime posizioni gli individui con energia idrofobica più bassa per ogni probabilist roadmap trovata.

Al termine di questo processo, la popolazione è classificata in maniera corretta per essere selezionata per la riproduzione dall'oggetto algoritmo genetico, così la classe GAPopulation può lavorare normalmente senza ridefinire la classe GASelectionScheme.

7.4.3 main

Il programma main, crea semplicemente l'oggetto algoritmo genetico definito da GAlib e gestisce la sua evoluzione. L'oggetto GAGeneticAlgoritm creato è il GASimpleGA, il quale richiama per default tutte le classi di GAlib giù definite, ad eccezzione delle due precedentemente descritte. I parametri settati per l'oggetto GASimpleGA sono i seguenti : popsize = 150, ngen = 200 , pmut = 0.7 , pcross = 0.3; questi sono stati scelti considerando anche i lavori già presenti che riguardavano il proteing folding su modello HP risolto grazie ad un algoritmo genetico[11, 21, 32, 33, 34, 38]. Il valore elevato di mutazione è giustificato dal fatto, la mutazione in questo lavoro, come già spiegato in precedenza, ha gli effetti di un

crossover. Mentre la probabilità di crossover è volutamente bassa, in quanto operazione ad elevato coefficiente di risultati non validi, ovvero con le strutture risultante non sono valide.

Una volta creato l'oggetto GASimpleGA, il programma main provvede alla sua evoluzione attraverso la funzione evolve, che richiama sia le classi standard, sia quelle implementate precedentemente. Da notare che l'inserimento degli nuovi individui è fatto rimpiazzando i "peggiori" padri. Inoltre il programma provvede alla visualizzazione su riga di comando del risultato dell'elaborazione.

Capitolo 8

Simulazione e Risultati

Per la fase di test del software realizzato, sono stati usati dieci benchmark standard di lunghezza 48 per il 3D protein folding HP. Le stringhe HP sono indicate nella tabella 8.1 e sono stati ampiamente utilizzati in letteratura [39, 40, 41, 42, 43].

I risultati degli esperimenti su queste stringhe HP che non riguardano l'algoritmo implementato sono quelli riportati da Fidanova in [44].

L'algoritmo dopo 200 iterazioni termina, così come riportato nel paper di Fidanova. Tutti gli esperimenti di Fidova sono stati eseguiti su IBM ThinkPad Centrino 1.8GHz CPU, 512 MB di RAM in esecuzione SuSe Linux, mentre l'algoritmo implementato in questo lavoro è stato eseguito su ADM Athlon(tm) XP2.5GHz CPU, 768 MB di RAM in esecuzione su Linux Ubuntu 9.10 .

Nella tabella 8.2, vengono mostrati i risultati ottenuti dai diversi dalle diverse euristiche. Per ognuna delle istanze di riferimento, è stato riportato il miglior risultato ottenuto. Per quanto riguarda le euristiche di Fidanova, sono state eseguite 20 volte, mentre nel nostro caso solamente 5.

Name	HP String
String1	HPHHPPHHHHPHHHPPHHPPHPHHPHPHHPPHHPPPHPPPPPPPPHHP
String2	HHHHPHHPHHHHHPPHPPHHPPHPPPPPPHPPHPPPHPPHHPPHHHPH
String3	PHPHHPHHHHHHPPHPHPPHPHHPHPHPPPHPPHHPPHHPPHPHPPHP
String4	PHPHHPPHPHHHPPHHPHHPPPHHHHHPPHPHHPHPHPPPPHPPHPHP
String5	PPHPPPHPHHHHPPHHHHPHHPHHHPPHPHPHPPHPPPPPPHHPHHPH
String6	HHHPPPHHPHPHHPHHPHHPHPPPPPPPHPHPPHPPPHPPHHHHHHPH
String7	PHPPPPHPHHHPHPHHHHPHHPHHPPPHPHPPPHHHPPHHPPHHPPPH
String8	PHHPHHHPHHHHPPHHHPPPPPHPHHPPHHPHPPPHHPHPHPHHPPP
String9	PHPHPPPPHPHPHPPHPHHHHHHPPHHHPHPPHPHHPPHPHHHPPPPH
String10	PHHPPPPPPHHPPPHHHPHPPHPHHPPHPPHPPHHPPHHHHHHHPPHH

Figura 8.1 3D-HP Benchmark

Name	HZ	CHCC	GG	CI	PERM	ACO-H	ACO-F	MyGA
String1	31	32	32	32	32	32	35,15	**32 (69.3)**
String2	32	34	34	33	34	34	36	**34 (71)**
String3	31	34	34	32	34	34	32,6	**34 (65.5)**
String4	30	33	33	32	33	33	40,6	**33(68.5)**
String5	30	32	32	32	32	32	35,15	**32 (70)**
String6	29	32	32	30	32	32	32,75	**32 (64.8)**
String7	29	32	32	30	32	32	33,8	**32 (63.5)**
String8	29	31	31	30	31	31	32,95	**31 (64.8)**
String9	31	33	33	32	34	34	34,44	**34 (67.3)**
String10	33	33	33	32	33	33	36,45	**33 (69.5)**

Figura 8.2 Confronto Risultati

HZ = Algoritmo Hydrophobic Zipper
CHCC = Algoritmo Constrain-based Hydrophobic Core Construction
CG = Algoritmo Chain Growth
CI = Algoritmo Contact Interactions
PERM = Algoritmo Pruned-Enriched Rosenbluth Method
ACO-H = Algoritmo ACO di Hoos
ACO-F = Algoritmo ACO di Fidanova
MyGA = Algoritmo genetico implementato in questo lavoro

Nella tabella 8.2 nella colonna MyGA sono riportati i risultati ottenuti dall'esecuzione del software realizzato. Tra parentesi è riportato il valore di energia idrofobica calcolato con il modello energetico riportato nel paper di Berenboym I. e Avigal M.[16], mentre l'altro valore è relativo al valore energetico contando solo i contatti idrofobici adiacenti.

Nella tabella 8.3, sono riportate le sequenze di mosse della sequenza amminoacidica che descrivono la conformazione tridimensionale ottimale del nucleo idrofobico raggiunta dal software.

Name	Sequenza di mosse (Coordinate Relative)
String1	RFFRBULBULDFFFRBUFLBBRRFRBBLDDRUFDFULFURDDLLLBB
String2	LBBRFDBLFLUFLBBBRFDDRFRBBUULDDLULUFUFDRFLURRDRU
String3	LFUBUFFLBDFDLUUBBUFFRRRDDLDRBUBUFUBLFLBDDRFDBRF
String4	RRFLLDFURDDFRBBBULLDRFURFUFRBDBUBDDFFFULLULDDBB
String5	RFLFDBBRFFFLURBRDDLBRUURBLDDLLFFFRRRBBBUFFFLURB
String6	RFFRBDDDLFUBBULDFFDBBRRRFFLURBBLUURDFUFDLLBLFUB
String7	RRRFULDLUULFURDDLDBUBRULUFRBRRDDLUFRULFRDLDRDLL
String8	RRRFLUUFDRDFULLBLBBUFFRBDDLFRRFLLUURRRBBDBLLURR
String9	RRFFRUULDLLUUBRFDBRDBRDFUUBUFFLBBDLULDDRFDFLBUU
String10	RBLBUUUFRFLDBRRFRBBUFFLBBLDRDLDRFRBUFFDLUBLLFRU

Figura 8.3 Configurazioni 3D risultati del software

Capitolo 9

Conclusioni e Sviluppi Futuri

Dalle prove effettuate, si può notare che i risultati ottenuti l'algoritmo implementato sono buoni, ovvero in linea con i risultati ottenuti dagli altri algoritmi, così come mostrato in tabella 8.2. Purtroppo considerazioni che riguardano una migliore conformazione del nucleo idrofobico rispetto agli altri algoritmi non si possono fare, in quanto non sono stati trovati dei risultati ottenuti con il modello energetico Global Energy[16]. Anche se possiamo sicuramente affermare che l'algoritmo trova un nucleo idrofobico, al peggio, uguale agli altri algoritmi.

Per quanto riguardano i lati negativi riscontrati dall'utilizzo di questo algoritmo, sicuramente vi è il tempo di esecuzione, che è abbastanza elevato, in comparazione con l'esecuzione degli altri algoritmi. Mediamente la fase di esecuzione, eseguita sulle stringhe benchmark di figura 8.1, oltrepassava l'ora di elaborazione. Questo è dovuto alle due fasi di ricerca nello spazio che l'algoritmo implementa : la ricerca nello spazio delle conformazioni (per la creazione di una configurazione valida, per l'operatore crossover e per l'applicazione delle pull moves) e la ricerca nello spazio energetico (per la costruzione delle probabilistic roadmaps) sono tutte fase molto onerose in termini computazionali.

Tuttavia, grazie allo studio dell'energy landscape con una semplicemente modifica dell'algoritmo si potrebbe ottenere, oltre che alla configurazione tridimensionale del nucleo idrofobico ottimale, anche il percorso ottimo che la proteina esegue per raggiungere lo stato nativo, partendo dallo stato denaturato. Il percorso di ripiegamento della proteina sarebbe a “costo zero” in termini di costi computazionali, in quanto non si necessiterebbero di ulteriori studi sull'energy landscape.

Inoltre utilizzando un pacchetto software reso disponibile dall'Università di Friburgo (lo stesso team che ha implementato la libreria BIU), sarebbe possibile ricostruire la struttura proteica tridimensionale su uno spazio tridimensionale libero (*off-lattice*) partendo dalla conformazione del nucleo idrofobico su reticolo cubico.

La fase di test, purtroppo, non è stata molto approfondita. Sarebbe interessante vedere se all'aumentare della sequenza amminoacidica, l'algoritmo proposto può avvicinarsi alle prestazioni degli altri algoritmi e anche è interessante effettuare un confronto con i tempi di

esecuzione con gli algoritmi che calcolano i percorsi del ripiegamento proteico, per poter dare una valutazione più completa sull'effettivo costo computazionale dell'algoritmo proposto.

Bibliografia

1. Bourne P.E. e Weissig H. (2003), "Structural bioinformatics", *Wiley*.

2. Dill K.A. (1990), "Dominant Forces in Protein Folding", *Biochemistry*, vol 29, no. 31, pp.7131-7155.

3. Anfinsen C.B., Haber E., Sela M. e White F.H. Jr (1961), "The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain", *Proceedings of the National Academy of Science*, vol. 47, no.9.

4. Levinthal, C. (1968), Journal Chimical Phyical 65, 44-45.

5. Dill K.A. e Chan H.S. (1997), "From Levinthal to pathways to funnels", *Nature Structural & Molecular Biology*, 4:10-9.

6. Anfinsen C. (1972), "The formation and stabilization of protein structure", *Biochemistry Journal*, vol. 128, no.4, pp.737–749.

7. Krasnogor N., Pelta D., Martinez Lopez P.E. e de la Canal E. (1998), "Genetic Algorithm for the Protein Folding Problem, a Critical View", *Engineering of Intelligent System*, ICSC Academic Press, pp. 353-360.

8. Paterson M. e Prytycka T. (1995), "On the Complexity of String Folding", *Proceedings of the Sixth International Conference on Genetic Algorithms*, 574-581.

9. Dill K.A. (1985), "Theory of the folding and stability of globular proteins", *Biochemistry,* 24:1501.

10. Almeida C.P., Gonçalves R.A., Goldbarg M.C., Goldbarg E.F.e Delgado M.R. (2007),

"TA-PFP: A Transgenetic Algorithm to Solve the Protein Folding Problem", *Seventh International Conference on Intelligent Systems Design and Applications*, pp.163-168.

11. Hoque M.T., Chetty M. e Dooley L.S. (2006), "A Guided Genetic Algorithm for Protein Folding Prediction Using 3D Hydrophobic-Hydrophilic Model", *WCCI / IEEE Congress on Evolutionary Computation*.

12. Hoque M.T., Chetty M. e Sattar A. (2007), "Protein Folding Prediction in 3D FCC HP Lattice Using Genetic Algorithm", *IEEE Congress on Evolutionary Computation*, pp.4138 – 4145.

13. Cutello V., Nicosia G., Pavone M. e Timmis J. (2007), "An Immune Algorithm for Protein Structure Prediction on Lattice Models", *IEEE Transactions on Evolutionary Computation*, vol.11, no.1, pp.101-117.

14. Kasnogor N., Hart W.E., Smith J. e Pelta D.A (1999), "Protein structure prediction with evolutionary algorithms", *Proc. Genetic Evol. Comput. Conf. Orlando*, pp.1596-1601.

15. Köing R. e Dandekar T. (2001), "Solvent entropy-driven searchin for protein modeling examined and tested in simplified models", *Protein Engineering*, vol.14 no.5 pp.329-335.

16. Berenboym I. e Avigal M. (2008), "Genetic Algorithms with Local Search Optimization for Protein Prediction Problem", *Proceedings of the 10th annual conference on Genetic and evolutionary computation* , pp.1097-1098.

17. Chen H., Zhou X., Liaw C.Y. e Koh C.G. (2004),"Kinetic Analysis of Protein Folding Lattice Models", *Modern Physics Letters B*, vol. 18, no. 4, pp. 163-172.

18. Cieplak M., Hoang T. X. e Li M.S. (1999), Phys. Rev. Lett., 83 ,1684.

19. Gutin A.M., Abkevich V. I. e Shakhnovich E.I. (1996), Phys. Rev. Lett., 77 , 5433.

20. Holland J. (1975), "Adaptation in Natural and Artificial Systems", *University of Michigan Press,* Ann Arbor.

21. Goldberg D.Edward. (1989), "Genetic algorithms in search, optimization, and machine learning", Addison-Wesley Longman Publishing Co., Inc .

22. Hirst J.D. (1999), "The evolutionary landscape of functional model proteins", *Protein Engineering*, vol.12 no.9 pp.721-726.

23. Sharma V., Kaila V. R. I. e Annila, A. (2009). "Protein folding as an evolutionary process". *Physica*, vol. 388, no.6, pp. 851–862.

24. S.Thomas, G.Song e N.M.Amato (2005),"Protein folding by motion planning", *Physical Biology*,vol. 2, pp. 148–155.

25. Bryngelson J.D. e Wolynes P.G. (1989), "Intermediates and barrier crossing in a random energy model (with applications to protein folding)", *J. Phys. Chem*, vol.93, pp.6902–6915.

26. Gruebele M. (2002), "Protein folding: the free energy surface", *Current Opinion in Structural Biology*, Vol.12, pp.161-168.

27. Kachalo S., Lu H. e Liang J., "Protein Folding Dynamics via Quantifications of Kinematic Energy Landscape"

28. Cho S.S., Levy Y. e Wolunes P.G. (2005),"P versus Q: Structural reaction coordinates capture protein folding on smooth landscapes", *Proceedings of the National Academy of Science*, vol. 103, no.3, pp.586-591.

29. Tang X. (2007), "Techniques for Modeling and Analyzing RNA and Protein Folding

Energy Landscapes", *Ph.D. Thesis*, Department of Computer Science and Engineering, Texas A&M University.

30. Song G. e Amato N.M. (2000), "A Motion Planning Approach to Folding : From Paper Craft to Protein Folding", *Technical Report TR00-017,* Department of Computer Science Texas A&M University.

31. Goodman J.E., O'Rourke J. e Indyk P. (2004), "Nearest neighbors in high-dimensional spaces", Chapter 39, *Handbook of Discrete and Computational Geometry* (2nd ed.).

32. Unger R. e J. Moult (1993), "Genetic algorithms for protein folding simulations", *Journal of Molecular Biology*, 231:75-81.

33. Song J., Cheng J., Zheng T. e Mao J. (2005), "A Novel Genetic Algorithm for HP Model Protein Folding", *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, pp. 935-937.

34. Custódio F.L., Barbosa H.J.C. e Dardenne L.E. (2004),"Investigation of the three-dimensional lattice HP protein folding model using a genetic algorithm", *Genetics and Molecular Biology,* vol. 27, no. 4, pp.611-615.

35. Lesh N., Mitzenmacher M. e Whitesides S. (2003), "A complete and effective move set for simplified protein folding", *Proceeding of the seventh annual international conference on Research in computational molecular biology*, ACM Press, pp.188-195.

36. Voβ S. e Woodruff D. (2002), "Optimization Software Class Libraries", Kluwer Academic Publishers, CAP.10 pp.295–329.

37. Mann M., Hamra M.A., Steinhöfel K. e Backofen R. (2009), "Constraint-based Local Move Definitions for Lattice Protein Models Including Side Chains", *Proceedings of the Fifth Workshop on Constraint Based Methods for Bioinformatics*.

38. Schulze-Kremer S. e Tiedemann U. (1994), "Parameterizing Genetic Algorithms for Protein Folding Simulation", *Proceedings of the Twenty-Seventh Annual Hawaii International Conference on System Sciences*, pp.345-354.

39. Beutler T. e K. Dill (1996), "A Fast conformational Method: A New Algorithm for Protein Folding Simulations", *Protein Sci*., no. 5, pp.147-153.

40. Hsu H. P., V. Mehra, W. Nadler e P. Grassbergen (2003), "Growth Algorithm for Lattice Heteropolymers at Low Temperature", *Chemical Physics*, no.118, pp.444-451.

41. Liang F. e W. H. Wong (2001), "Evolutionary Monte Carlo for Protein Folding Simulations", *Chemical Physics*, 2001, no.115, pp. 444-451.

42. Shmygelska A. e H. H. Hoos (2005), "An Ant Colony Optimization Algorithm for the 2D and 3D Hydrophobic Polar Protein Folding Problem", *BMC Bioinformatics*, vol.6, no.30.

43. Toma L. e S. Toma (1996), "Contact Interaction Method: A New Algorithm for Protein Folding Simulations", *Protein Sci.*, 1996, no.5, pp.147-153.

44. Fidanova S.(2006), "3D HP Protein Folding Problem using Ant Algorithm", *BioPS'06, October 24-25,* III.19-III.26.

Rifermenti Internet

I1. http://it.wikipedia.org/wiki/Evoluzione