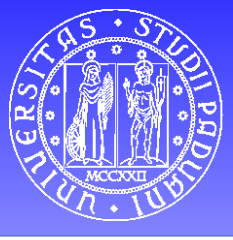
Bigger Python

Progetto di Intelligenza Artificiale

Autori: Andrea Grendene, Davide Colacicco



BiGGER: paper originale

- Lo scopo principale del paper originale era di costruire un algoritmo per la risoluzione del problema del docking usando la Programmazione con vincoli
- Il miglior metodo usato fino a quel momento si basava sulle FFT (Trasformate di Fourier veloci)
- La Programmazione con vincoli dovrebbe permettere di risolvere il problema con una complessità di tempo simile
- Ma richiede molta meno memoria ed è più flessibile

BiGGER: il nostro progetto

- Il nostro scopo è capire come viene utilizzata la Programmazione con vincoli in BiGGER
- Dato che l'applicazione è abbastanza complessa ed è scritta in Pascal abbiamo deciso di riprodurla in Python
- Questo ci ha permesso di comprendere meglio come funziona Pascal, la struttura originale del codice e le scelte effettuate dagli autori originali
- In più abbiamo potuto fare un confronto di prestazioni tra Pascal e Python

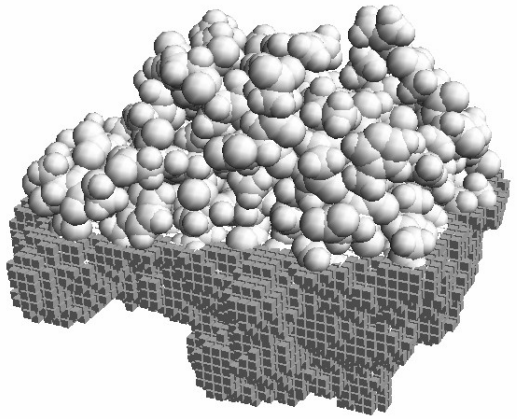
Caricamento delle proteine

- Ogni proteina viene caricata da un file PDB
- È un tipo di file apposito, che contiene tutte le informazioni relative alla struttura della proteina e agli atomi singoli
- Ai fini dell'applicazione vengono usati soltanto le coordinate e i raggi degli atomi
- Successivamente ogni proteina viene caricata in una griglia apposita

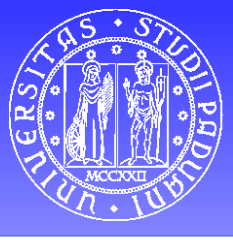
Costruzione della griglia della proteina

- Viene costruita una prima griglia tridimensionale di supporto, riempita di zeri, in cui la lunghezza dei lati delle celle è pari al raggio massimo degli atomi

- Le dimensioni di questa griglia sono pari alla differenza fra il valore massimo e quello minimo delle rispettive componenti delle coordinate degli atomi, divisa per la lunghezza del lato della cella



- Per ogni atomo viene poi trovata la cella che contiene la sua coordinata (adattata alla griglia), quindi nella cella viene salvato l'indice dell'atomo



Costruzione della griglia della proteina

- Viene costruita una seconda griglia, sempre tridimensionale, in cui le prime due dimensioni hanno lunghezza pari alla differenza tra il valore massimo e quello minimo della rispettiva componente delle coordinate degli atomi

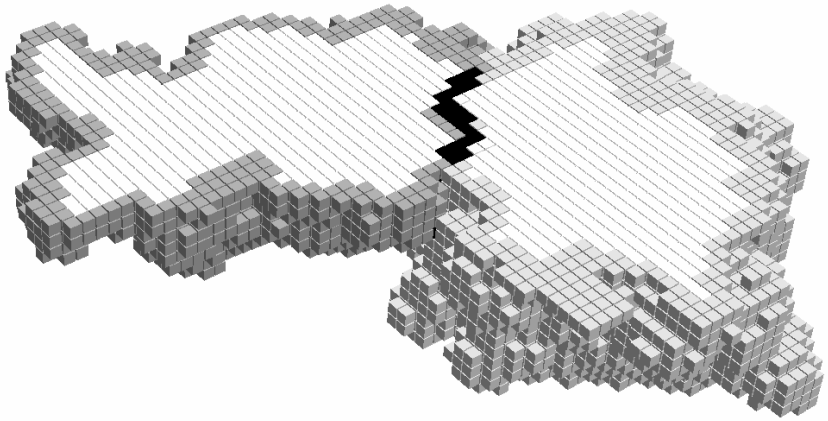
- La terza dimensione invece contiene degli intervalli, che rappresentano le celle occupate dagli atomi

- Viene ricavata la terza dimensione come descritto sopra per le altre due, quindi per ogni cella di questa griglia viene verificato se la sua coordinata si trova all'interno del raggio di un atomo tramite la griglia costruita in precedenza

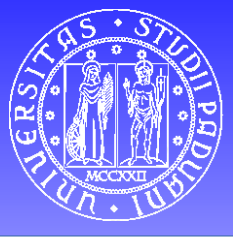
Costruzione della griglia della proteina

- Se è così viene impostato il corrispondente valore di un array temporaneo a 1, quindi le celle contigue di questo array che hanno valore pari a 1 formeranno gli intervalli per la X e la Y in analisi

- Infine gli intervalli vengono divisi tra due ulteriori griglie, che conterranno rispettivamente le celle appartenenti alla superficie e al nucleo della proteina



- Per ognuna di queste griglie viene anche costruito un altro array, in cui sono contenuti gli intervalli delle Y che contengono almeno un intervallo in Z

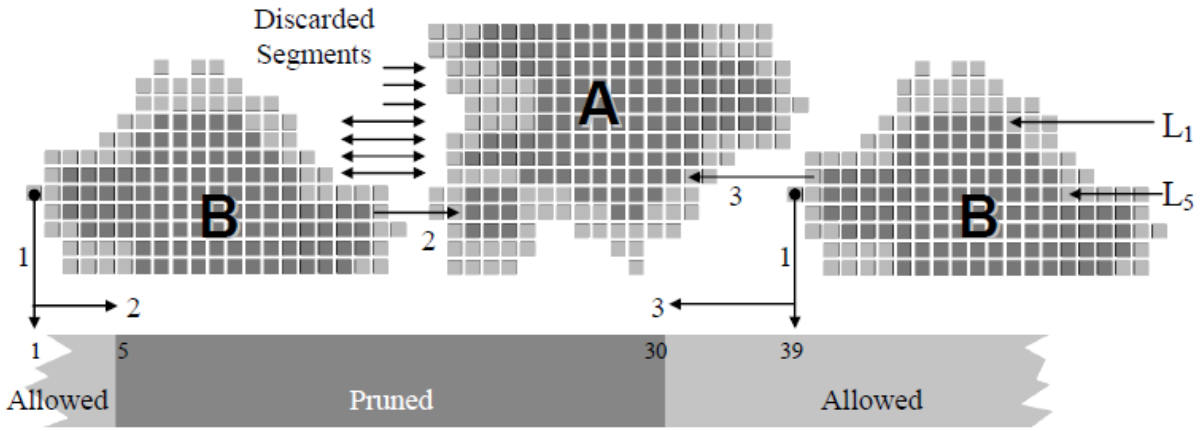


Applicazione della Programmazione con vincoli

- Per simulare l'interazione tra le proteine viene costruita un'altra griglia, le cui dimensioni sono pari alla somma di quelle delle griglie delle due proteine, che costituiscono oltretutto il dominio del problema

- Lo scopo dell'applicazione della Programmazione con vincoli sarà ridurre il più possibile questi domini

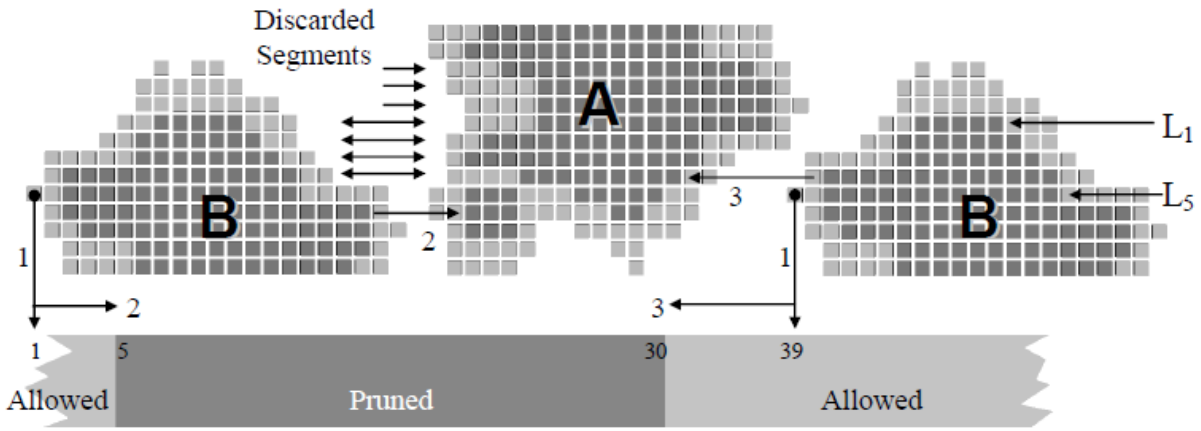
Applicazione della Programmazione con vincoli



- Per tutti gli X di questa griglia viene verificato se per ogni intervallo in Y delle due proteine ci sono intersezioni, e per ogni coppia di segmenti viene preso l'intersezione di lunghezza maggiore; quindi le intersezioni costituiranno gli intervalli in Y della griglia generale

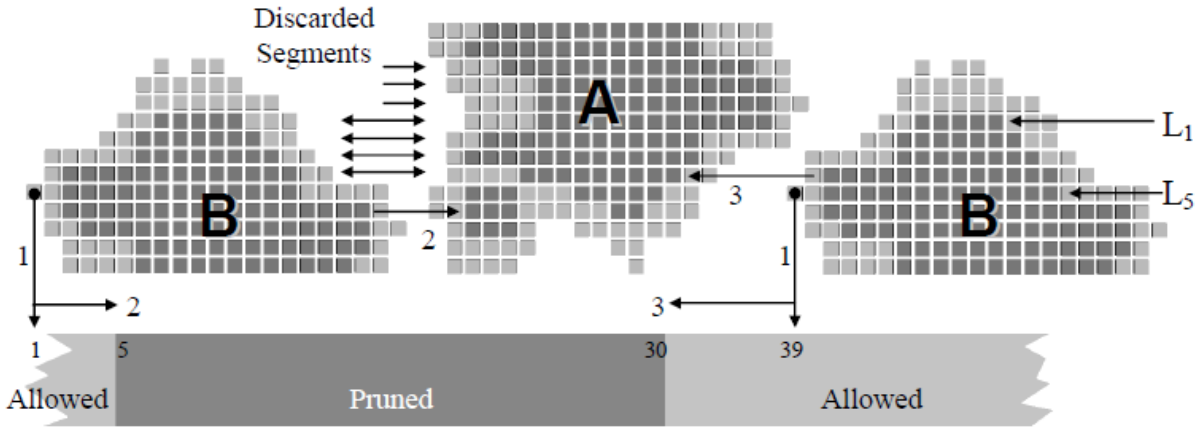
- Questi sono i primi due vincoli del problema

Applicazione della Programmazione con vincoli



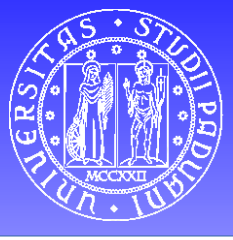
- Successivamente vengono trovate le sovrapposizioni degli intervalli in Y che appartengono al nucleo delle proteine
- Tali intervalli vengono eliminati da quelli ricavati nel punto precedente
- Questo è il terzo vincolo: non ci possono essere intersezioni che coinvolgono celle appartenenti al nucleo

Applicazione della Programmazione con vincoli



- I passaggi appena visti vengono ripetuti con gli intervalli in Z contenuti negli intervalli in Y rimasti

- Inoltre per ogni coppia di intervalli in Y viene calcolato un punteggio, basato sul numero di celle delle sovrapposizioni in Z trovate



Applicazione della Programmazione con vincoli

- Applicando la tecnica del Branch and Bound, se il punteggio trovato è minore di un valore minimo fissato allora gli intervalli vengono svuotati, e quindi tolti dal dominio

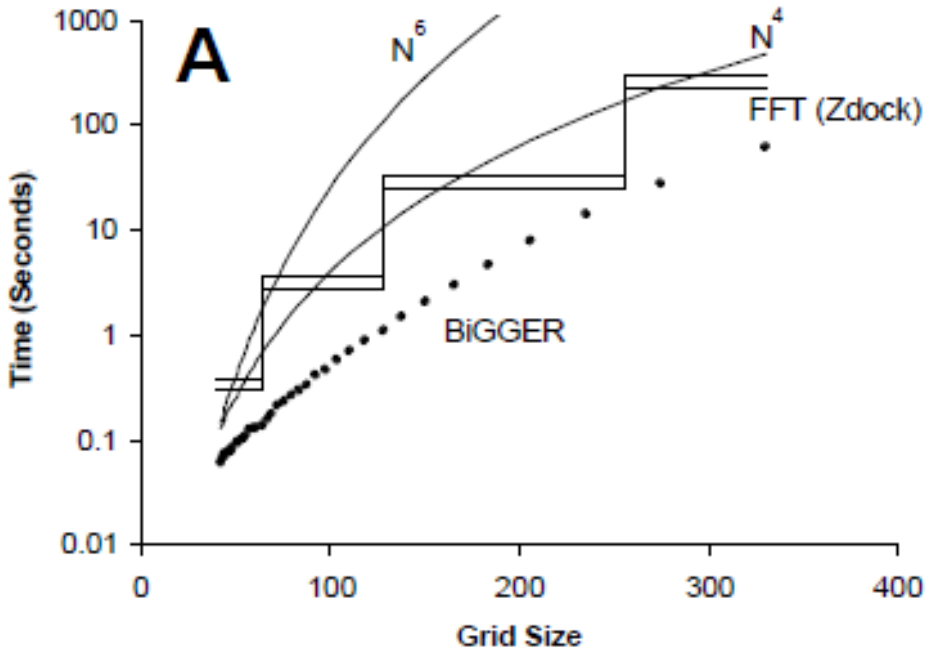
- Infine per calcolare i punteggi si contano le celle che possono sovrapporsi per ogni intervallo in Y rimasto, quindi il risultato finale sarà composto dal punteggio maggiore e dalla coordinata in cui inizia la sovrapposizioni dei due intervalli migliori

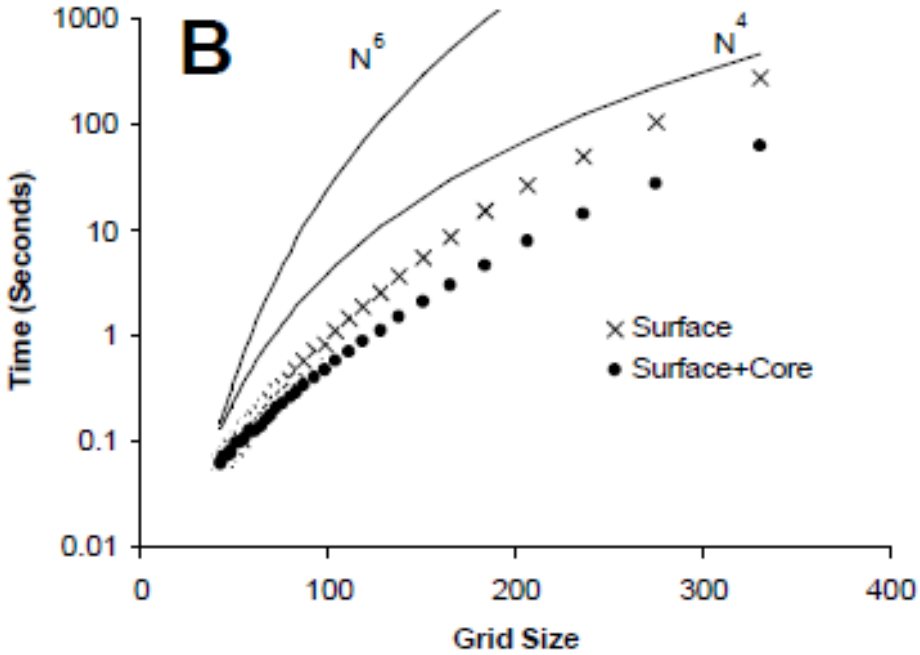
Risultati

	Python PDB (secondi)	Pascal PDB (secondi)	Python BiGGER (secondi)	Pascal BiGGER (secondi)
Risultati (media di 10 esecuzioni)	0.3719	0.1109	251.6822	3.7982

	Algoritmo generale	FFT	BiGGER
Complessità di tempo	$O(N^6)$	$O(N^3 \log N)$	$O(N^4)$
Complessità di memoria	-	$O(N^3)$	$O(N^2)$

Risultati


A
1000
100
10
1
0.1
0.01
Time (Seconds)
0
100
200
300
400
Grid Size
N6
N4
FFT (Zdock)
BiGGER


B
1000
100
10
1
0.1
0.01
Time (Seconds)
0
100
200
300
400
Grid Size
N6
N4
Surface
Surface+Core