Wintersemester 2018/2019, GST

Übung: Berechnung oberer und unterer Grenzen mit Floyds Algorithmus

Abgabe der Lösung bis 18.11.2018

1 Floyds Algorithmus

Die intramolekularen Distanzen, die man aus einem NMR-Experiment bekommt, sind meist nicht vollständig. Man kann die Koordinaten der Molekülstruktur also nicht direkt berechnen. Der erste Schritt zum Berechnen von Proteinkoordinaten aus Distanzen ist deshalb oft die Bestimmung oberer und unterer Grenzen. Der Berechnung dieser Grenzen liegt die Dreiecksungleichung

$$d_{ij} \leq d_{ik} + d_{kj} \tag{1}$$

oder auch

$$|d_{ik} - d_{kj}| \leq d_{ij} \tag{2}$$

zugrunde. Alle oberen Grenzen lassen sich mit Floyds Algorithmus folgendermaßen berechnen:

```
for k in 1 to N
    for all pairs i and j
        d[i,j] = min(d[i,j], d[i,k] + d[k,j])
```

d[i,j] beinhaltet dabei die aktuelle obere Grenze für die Distanz zwischen den Atomen i und j. Untere Grenzen lassen sich ebenfalls mit Floyds Algorithmus unter Berücksichtigung der in Gleichung 2 gezeigten Form der Dreiecksungleichung berechnen.

2 Implementierung

Ziel dieser Übung ist die Implementierung von Floyds Algorithmus zur Berechnung der oberen und unteren Grenzen für fehlende Distanzen. Es darf jede beliebige (unter Linux kompilierbare bzw. ausführbare) Programmiersprache verwendet werden.

Alle Dateien befinden sich in `/home/petersen/teaching/floyd`. Kopieren Sie sich diesen Ordner am besten in ein lokales Verzeichnis.

2.1 Für Programmieranfänger

Wer noch nicht so viel Erfahrung im Programmieren hat kann einen in Python geschriebenen Rahmen benutzen. Darin sind bereits alle Ein- und Ausgaberoutinen implementiert und es müssen nur die Schleifen von Floyds Algorithmus eingefügt werden.

Der Code befindet sich in den Dateien `floyd.py`, `distances.py` und `test_floyd.py`. `floyd.py` enthält die `main`-Funktion sowie Routinen zum Einlesen und zur Ausgabe des Ergebnis. In der Funktion `floyd_algorithm` muss der fehlende Code eingefügt werden. In `distances.py` befindet sich die Klasse `Distances`, welche eine symmetrische Distanzmatrix implementiert. Wichtig zu wissen ist, dass man die Distanz zwischen i und j in der Matrix `dist` (Beispielname) mittels `dist[i,j]` bekommt und mit `dist[i,j] = val` auf den Wert `val` setzen kann.

Das Program lässt sich folgendermaßen aufrufen:

```
python floyd.py 1L2Y_100.key
```

wobei `1L2Y_100.key` durch jede andere Eingabedatei ersetzt werden kann. Die Ausgabe wird in die Dateien `*_upper.txt` und `*_lower.txt` geschrieben. `*` steht hier für das Präfix des Namens der Eingabedatei.

Hat man die Routine um die oberen oder die oberen und unteren Grenzen erweitert, lässt sich außerdem mit

```
python test_floyd.py -v
```

ein einfacher Test ausführen. Das Skript testet die oberen und unteren Grenzen separat.

2.2 Eingabe

Das Eingabeformat soll den im Tinker Packet verwendeten .key Dateien entsprechen. Mit dem Programm keyfilegen lassen sich diese Dateien aus Distanzdateien (.dis) erzeugen (siehe Übungszettel 2).

Eine Beispieldatei befindet sich in `/home/petersen/teaching/floyd/1L2Y_100.key`. Öffnen Sie die Datei und sehen Sie selbst. In Spalte 2 und 3 findet man die beiden Indices (starten mit 1) der Atome und in den Spalten 4 und 5 die untere und die obere Grenze für die jeweilige Distanz. Weitere Beispiele befinden sich im Ordner `/home/petersen/teaching/floyd`.

2.3 Ausgabe

Das Ergebnis soll in zwei Dateien geschrieben werden, welche jeweils eine halbe Matrix mit Distanzen enthalten. Eine Datei entält die oberen Grenzen, eine zweite die unteren. Dabei soll die untere Hälfte der Matrix **ohne Diagonale** verwendet werden. In den Dateien sollen Fließkommazahlen stehen, die von Leerzeichen getrennt werden. Ein Beispiel mit 4 Atomen:

```
1.48
2.52 4.00
3.43 4.91 5.95
```

Diese Ausgabe kann dann auch wie in Abschnitt 2.5 beschrieben visualisiert werden.

2.4 Der Algorithmus

Die oberen Grenzen können genau wie in Abschnitt 1 berechnet werden. Bei der Implementierung ist lediglich darauf zu achten, die oberen Grenzen aus der Distanz-Datei (.key) zu verwenden.

Die Berechnung der unteren Grenzen ist ein kleines bißchen komplizierter. Hier müssen die zuvor berechneten oberen Grenzen sowie die mit der Eingabedatei übergebenen unteren Grenzen berücksichtigt werden. Außerdem ist die Dreicksungleichung wie in Gleichung 2 dargestellt anzuwenden. Es kann sinnvoll sein eine Skizze mit Dreieck, Minima und Maxima zu zeichnen.

2.5 Visualisierung des Ergebnis

Bei der Auswertung der Ergebnisse hilft das Skript `boundsvis.py`. In Form einer Heatmap werden die berechneten Distanzen oder die Unterschiede zwischen oberen und unteren Grenzen dargestellt. Das Skript bekommt als Eingabe entweder einen oder zwei Namen von Dateien mit Distanzmatrizen und optional einen Grenzwert `max_dist` üebergeben.

Die Farbe der 2-dimensionalen Heatmap an Position (i,j) ist abhängig von der Distanz zwischen den Atomen i und j. Die Verteilung der Farben hängt vom gewählten Wertebereich ab und kann anhand der gezeigten Legende nachvollzogen werden. Um die Ergebnisse zweier unterschiedlicher Fälle miteinander vergleichen zu können, muss dementsprechend derselbe Wertebereich gewählt werden. Dafür benutzt man den Parameter `max_dist`.

Man könnte das Skript also folgendermaßen ausführen:

```
python boundsvis.py 1L2Y_100_lower.txt --max_dist 100
```

In diesem Fall wurde nur eine Distanzmatrix übergeben. Dementsprechend stellt die Abbildung die in der Datei gespeicherten Grenzen dar. Interessanter kann es sein, die Unterscheide zwischen oberen und unteren Grenzen zu betrachten. Dafür kann man einfach eine zweite Datei mitgeben, z.B.

```
python boundsvis.py 1L2Y_100_lower.txt 1L2Y_100_upper.txt --max_dist 25
```

3 Anwendung

Probieren Sie ihr Programm aus und betrachten sie die Ergebnisse. In dem kopierten Verzeichnis befinden sich bereits einige Dateien die aus der Datei `1L2Y.dis` erzeugt wurden. Im Dateinamen ist der Anteil der verwendeten Distanzen in % enthalten. Mit dem Programm `keyfilegen` (`/home/petersen/teaching/distgeom/keyfilegen`) können weitere Beispiele erzeugt werden (siehe Übungszettel 2). Wie verändern sich die Grenzen, wenn man unterschiedliche Mengen von Distanzen verwendet oder die Unsicherheit der Distanzen erhöht?

4 Abgabe

Schickt euren Code an **petersen@zbh.uni-hamburg.de**.