Andrew Torda
Timur Olzhabaev

Zentrum für Bioinformatik
Übung zur Vorlesung
Angewandte Bioinformatik: Strukturen
Wintersemester 2017/2018

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

16. Januar 2018

Übung 8: Simulated Annealing

Im Folgenden wird anhand eines abstrakten Minimalbeispiels der „Simulated Annealing“ Optimierungs-Algorithmus vorgestellt. Im Kontext der Veranstaltung könnte der Algorithmus z.B. zur Protein- / RNA-Sequenzoptimierung genutzt werden. Um die Grundideen zu verstehen, werden hier jedoch einfache eindimensionale stetige Zielfunktionen betrachtet.

Es werden zwei Skripte verwendet: Ein ***Python*** Skript mit dem Algorithmus angewendet auf definierte Zielfunktionen und ein ***Gnuplot*** Skript, welches die Ergebnisse visualisiert.

Starten Sie eine Konsole und kopieren Sie den Ordner mit den Dateien zur Übung in Ihr Homeverzeichnis mit:

```
cp -R /home/olzhabaev/Public/annealing  .
```

Öffnen Sie das ***Python*** Skript **`annealing.py`** und verschaffen Sie sich einen Überblick. Relevant sind die Zeilen 20 bis 65. Hier passiert zunächst folgendes:

- Die Zielfunktion wird gewählt.
- Der Startwert (***x***) wird initialisiert.
- Die Parameter des Algorithmus werden festgelegt. Hierbei handelt es sich um:
  - Anzahl der Optimierungsschritte
  - Standardabweichung der durchschnittlichen Schrittlänge
  - Initiale Temperatur
  - Faktor um den die Temperatur in jeder Iteration gesenkt wird
- Es werden Listen angelegt, welche die besuchten ***x*** Werte und Temperaturen speichern.

In Zeilen 40 bis 64 ist der Algorithmus selbst definiert. Es handelt sich um einen Loop, der eine feste Anzahl von Iterationen läuft. In jeder Iteration wird ein Probeschritt ausgeführt, indem ein Nachbar von ***x*** gesampelt wird und geprüft wird, ob die Zielfunktion an dieser Stelle einen geringeren Wert hat. In diesem Fall wird der Schritt akzeptiert. Andernfalls wird der Wert nur mit einer gewissen Wahrscheinlichkeit abhängig von der Differenz zum

vorherigen Wert und der Temperatur akzeptiert. Anschließend wird geprüft, ob ein neuer Minimalwert gefunden wurde und dieser ggf. überschrieben. Schließlich wird die Temperatur um einen Faktor gesenkt und der Besuchte ***x***- und Temperaturwert den entsprechenden Listen zugefügt.

Weiter unten werden die gesammelten Ergebnisse in entsprechende Dateien geschrieben. Es wird dabei eine Datei für die in gleichmäßigen Schritten gesampelte Zielfunktion erstellt (**`objective_function_values.txt`**). Weiterhin wird eine Datei mit den durch den Algorithmus besuchten ***x***-, Zielfunktion- und Temperaturwerten erzeugt (**`visited_function_values.txt`**). Der beste gefundene ***x*** Wert wird mit entsprechendem Wert der Zielfunktion ebenfalls in eine Datei geschrieben (**`best_value.txt`**). Das ***Gnuplot*** Skript **`plot.gnu`** verwendet die erstellten Dateien um eine Visualisierung des Laufs des Algorithmus zu erstellen.

Wechseln Sie in der Konsole in das **`annealing`** Verzeichnis und lassen Sie die beiden Skripte laufen:

```
cd annealing
python annealing.py ; gnuplot plot.gnu
```

Öffnen Sie die erstellte Visualisierung **`output.pdf`** und betrachten Sie die Ergebnisse. Experimentieren Sie nun mit den Parametern des Algorithmus. Setzen Sie andere Werte ein (Zeilen 30 bis 33 - vergessen Sie nicht die Datei anschließend zu speichern) und lassen Sie das Skript mehrmals laufen. Versuchen Sie nachzuvollziehen, wie die Parameter die Optimierung beeinflussen. Sie können auch eine andere Zielfunktion auswählen (Zeile 23).

Aufgaben

Bitte beantworten Sie die folgenden Fragen möglichst kurz und schicken Sie Ihre Antworten an olzhabaev@zbh.uni-hamburg.de als Anhang im plain text oder pdf Format bis zum 22.01.2018, 08:00 Uhr. Fügen Sie dabei bitte das Stichwort `[AST]` dem Betreff zu.

a) Ist der „Simulated Annealing“ Algorithmus deterministisch? Wenn nein, an welchen Stellen werden im Algorithmus probabilistische Entscheidungen getroffen?
b) Garantiert der Algorithmus das globale Optimum zu finden? Warum bzw. warum nicht?
c) Wenn man anstatt einer Minimierung eine Maximierung durchführen möchte, was müsste man an dem Algorithmus oder der Zielfunktion ändern?
d) Der Algorithmus erzeugt in jeder Iteration einen zufälligen Schritt in der Zielfunktion. Wie würde dieser beim Protein- bzw. RNA-Design aussehen?