Übungsblatt: Lokale Suche und Genetische Algorithmen (GA)

Enes Pohland

26. Oktober 2025

Bonus: ChatGPT Taboo (2P)

Wir haben Tabu mit ChatGPT gespielt. Unser Zielwort war **Proteinbiosynthese**. Wir durften dabei eine Reihe von Wörtern nicht benutzen (z.B. Protein, mRNA, Translation, Transkription usw.). Ich habe versucht, das Wort durch umschreiben zu erklären, also über Begriffe wie „Bauplan“, „Zelle“ und „Kette von Bausteinen“.

Nach nur wenigen Sätzen hat ChatGPT das Wort tatsächlich erraten und „Proteinbiosynthese“ gesagt. Das ging erstaunlich schnell und war ziemlich einfach, ehrlich gesagt. Ich hab zuerst allgemein über Zellprozesse gesprochen, und er hat direkt gefragt ob ich die Proteinherstellung meine. Dann hab ich gesagt „ja, wie heißt das nochmal genau?“ und er hat es direkt gesagt. Also das war relativ leicht. Ich hätte gedacht das dauert länger.

Fazit: War ein netter Test, zeigt aber wie stark Sprachmodelle auf semantische Hinweise reagieren, selbst wenn man die wichtigsten Wörter weglässt. Schwierigkeit war gering.

—

EA.01: Modellierung von GA (2P)

(a) 8-Queens Problem

Kodierung: Ein Individuum ist eine Permutation von 1 bis 8. Die Position i steht für die Zeile, und der Wert x[i] ist die Spalte. Dadurch gibt es keine Konflikte in Zeilen oder Spalten, nur auf Diagonalen.

Fitness: $f = 28-$ Anzahl der diagonal Konfliktpaare (28 ist max. Anzahl möglicher Paare). Man kann auch einfach $f = -$Konflikte nehmen, aber Maximierung ist verständlicher.

Crossover: Partially Mapped Crossover (PMX) oder Order Crossover (OX), damit keine Spalten doppelt vorkommen.

Mutation: Swap zweier Werte oder Scramble eines kleinen Segments.

Selektion: Turnierselektion (k=2 oder 3) oder Rang-Selektion.

Für Simulated Annealing: Man braucht einen Startzustand (Permutation), eine Nachbarschaft (zwei Werte tauschen), eine Energie-Funktion (Anzahl Konflikte) und einen Temperaturplan. Beispiel: Start T=100, dann $T \leftarrow 0.95T$ jede Iteration.

—

(b) Landkarten-Färbung

Kodierung: Ein Individuum ist ein Vektor $x \in \{1, ..., K\}^n$, wobei jede Region eine Farbe kriegt. Anfangs 5 Farben.

Fitness: Konflikte zählen (benachbarte Regionen gleiche Farbe). Fitness = $-(\lambda \times Konflikte + aktiveFarben)$ mit λ groß (z.B. 1000). So werden erst Konflikte, dann Farbanzahl minimiert.

Crossover: Uniform oder Cluster-basiert (z.B. ganze Regionen übernehmen).

Mutation: Recolor einer Region auf andere Farbe, eventuell konflikt-basiert. Manchmal auch „Kempe-Chain"-Tausch.

Für Simulated Annealing: Zustand = Färbung, Nachbarschaft = eine Region umfärben, Energie = λ·Konflikte + Farben. Temperatur sinkt geometrisch.

—

EA.02: Implementierung (5P)

Algorithmus:

```
init population
evaluate
repeat until stop:
    select parents (tournament)
    crossover with prob pc
    mutate with prob pm
    evaluate offspring
    replace worst with best (elitismus)
```

Parameter, die getestet wurden:

- Populationsgröße: 30, 50, 100
- Crossoverrate: 0.7, 0.9, 1.0
- Mutationsrate: 0.05, 0.1, 0.2

- Elitismus: 0%, 2%, 5%

Metriken:

- Erfolgsrate (Prozent Läufe mit optimaler Lösung)
- Durchschnittliche Generationen bis Erfolg
- Durchschnittliche Fitness

Die Ergebnisse zeigten, dass für 8-Queens PMX + Swap und kleine Mutation (0.1) am besten funktioniert. Beim Map Coloring hat eine kleine „Repair"-Funktion nach Mutation sehr geholfen. Je größer die Population, desto stabiler, aber auch langsamer.

—

EA.03: Anwendungen (3P)

(1) Randal Olson – „Here's Waldo"

Nutzt GA um die optimale Suchstrategie für Waldo zu finden. Kodierung ist Position auf Karte, Fitness ist Suchzeit oder Wahrscheinlichkeit Waldo zu finden. Er nutzt auch heuristische Optimierung und Visualisierung.

(2) Evolution Simulator

Im „Evolution Simulator" von Keiwan werden Kreaturen als Körpergrafen mit Muskeln und Gelenken codiert. Mutationen verändern die Struktur oder Bewegungsparameter. Fitness = Distanz oder Geschwindigkeit. Sehr anschauliches Beispiel wie komplexe Körper mit GA entstehen können.

(3) american fuzzy lop (AFL)

AFL nutzt evolutionäre Ideen im Software-Fuzzing. Individuum = Eingabedatei, Mutation = Bit/Byte-Flips oder Blocktausch, Crossover = „Splicing" zweier Seeds. Fitness = neue Programm-Pfade oder mehr Code-Coverage. Sehr effektiver Ansatz.

Weitere Anwendungen

EAs werden auch bei Antennen-Design (z.B. NASA), Neural Architecture Search (NAS), und Scheduling eingesetzt. Immer geht es darum, in riesigen Suchräumen gute Lösungen zu finden, wo klassiche Verfahren schwer tun.

—

Fazit

Ich fand die Aufgaben insgesamt spannend. Der Teil mit ChatGPT war eher Spaß, aber der GA-Teil hat gut gezeigt, wie wichtig Kodierung und Fitnessgestaltung sind. Wenn man die richtig wählt, klappt die Optimierung erstaunlich gut.