Bericht zur Maturaarbeit

Simulated Annealing - Schwierige Probleme einfach gelöst

H.R. Schneider, Kantonsschule am Burggraben, St. Gallen

Ausgangslage

S, eine mathematisch interessierte und begabte Schülerin, besuchte in der dritten Klasse die von der ETH speziell für Frauen angebotene Studienwoche Informatik. Dort konnte S ihre Kenntnisse in Algorithmik und Programmieren weiter ausbauen und vertiefen. Die Freude am Lösen algorithmischer Probleme und das dazugehörige Selbstvertrauen bestimmten dann zusammen mit ihrer Vorliebe für Knobelspiele das Thema der Maturaarbeit.

Problembeschreibung

Der übliche Lösungsweg eines vorgegebenen Optimierungsproblems besteht darin, einen deterministischen Algorithmus zu entwerfen, dessen Korrektheit unter gewissen Vorbedingungen zu prüfen, ihn dann in einer Programmiersprache zu implementieren und durch Testen die durch Unachtsamkeit eingebauten Laufzeit- und (weit schlimmer) logischen Fehler auszumerzen. Für jedes „schwierige“ Problem muss so individuell eine Lösungsmöglichkeit entwickelt und diese in ein Programm umgesetzt werden. Man benutzt dazu verschiedene Programmierstrategien: Backtracking, dynamisches Programmieren, „Teile und Herrsche“-Strategie, Greedy-Methode etc. Die exakte Lösung eines Problems hat dementsprechend einen hohen Preis: Lange Entwicklungszeit und bei wirklich schwierigen Problemen zu lange Laufzeit.

Anders sieht es aus, wenn man den Zufall zu Hilfe nimmt. Gilt es im berüchtigten Heuhaufen die goldene Stecknadel zu finden, so nimmt man sich vor, z.B. eine Stunde lang oder bis zum Auffinden der Nadel, zufällig eine potenzielle Lösung herauszugreifen. Ist der Zugriff optimal, d.h. der Halm ist die goldene Nadel, so bricht man die Suche glücklich ab.

Dieses Verfahren hat den Nachteil, dass nicht immer (!) die Lösung gefunden wird. Der Vorteil des Verfahrens ist seine Universalität. An Stelle der Stecknadel im Heuhaufen kann auf prinzipiell dieselbe Weise eine Landkarte mit vier Farben gefärbt werden: Eine Färbung ist optimal, wenn keine zwei Länder mit gemeinsamer Grenze die gleiche Farbe besitzen.

Dieses grobe Verfahren lässt sich durch raffinierte Steuerung des Zufalls um Grössenordnungen verbessern. Simulated Annealing (simulierte Abkühlung) orientiert sich an einem physikalischen Phänomen: Lässt man ein stark erhitztes Metall langsam abkühlen, so geht der Zustand von hoher Energie und chaotischer, praktisch bindungsloser Teilchenstruktur in einen Zustand minimaler Energie über. Die Änderung der Lage jedes einzelnen Teilchens wäre nur mit etwas Energiezufuhr möglich. Der Endzustand ist also bezüglich der Gesamtenergie (zumindest lokal) optimal. Wendet man dieses Phänomen auf ein Optimierungsproblem an, tritt an die Stelle des momentanen Zustands des Metalls eine Anordnung der Probleminstanz. Weiter benötigt man eine Funktion, welche aus einer Anordnung A ihre Energie $E(A)$ berechnet. An Stelle der Zustandsänderung eines Metallteilchens tritt eine Tauschfunktion, welche aus der aktuellen Anordnung zufällig eine leicht veränderte Anordnung ermittelt. Die Gesamtheit dieser Funktionswerte, sozusagen die

Nachbarschaft einer Anordnung, entspricht dann den möglichen Zustandsänderungen aller Metallteilchen im beobachteten Zeitpunkt.

```
Universeller Algorithmus für Simulated Annealing

     Aneu, Aalt: Variablen für Anordnungen der betrachteten Probleminstanz
     T, Tstart,Tend : Variablen für die aktuelle, Start- und Endtemperatur
     k              : Änderungsfaktor zur Modellierung exponentieller
                      Temperaturabnahme, 0<k<1
     E: A  -> E(A)  : Energiefunktion, ist E(A)minimal, so ist A optimal
                      d.h. eine Lösung ist gefunden
     C: A -> C(A)   : Tauschfunktion, berechnet aus der Anordnung A eine
                      zufällige Anordnung in der "Nachbarschaft" von A
     na             : na etwas grösser als die Anzahl Elemente des
                      Wertebereichs von C(A) für festes A.
     rand: (a,b) -> rand(a,b): generiert eine gleichverteilte Zufallszahl
                      im Intervall [0,1]

     BeginSimulatedAnnealing
       T:=TStart;
       Generiere eine zufällige Anordnung in Aalt;
       While T>Tend Do
         For i:=1 To na Do
           Aneu:=C(Aalt);
           If (E(Aneu)<=E(Aalt)) or            {
              (exp(-1/T)> rand(0,1) Then
             Aalt:=Aneu;
           EndIf
         EndFor
         T:=T*k;
       EndWhile
     EndSimulatedAnnealing
```

Soll nun ein konkretes Problem mit Simulated Annealing gelöst werden, so muss zuerst die Datenstruktur für eine Anordnung definiert werden (Im einfachen Fall ein Vektor oder eine Matrix). Weiter ist die Tauschfunktion zu entwerfen und gleichzeitig *na*, eine Schätzung der Anzahl „Nachbarn“ einer Anordnung zu bestimmen. Kreativität erfordert die Festlegung der Energiefunktion: Diese soll möglichst wenig lokale Minima mit möglichst sanft ansteigenden Flanken haben. Schliesslich müssen die Parameter Tstart, Tend und der Verminderungsfaktor k durch Experimentieren bestimmt werden.

Ergebnisse und Erfahrungen

S hat in ihrer Maturaarbeit einerseits eine grosse Zahl von verschiedenartigen Problemen mit Hilfe von Simulated Annealing gelöst: Das verflixte Hexenpuzzle, Magisches Quadrat, N-Damenproblem, Lapland, Pentominopuzzle, Isomorphietest für Graphen und das Traveling Salesman Problem.

Andererseits hat sie in JAVA ein Rahmenprogramm entworfen bei dem, OOP sei Dank, nur die für ein konkretes Problem relevanten Klassen mit ihren Methoden zu programmieren sind.

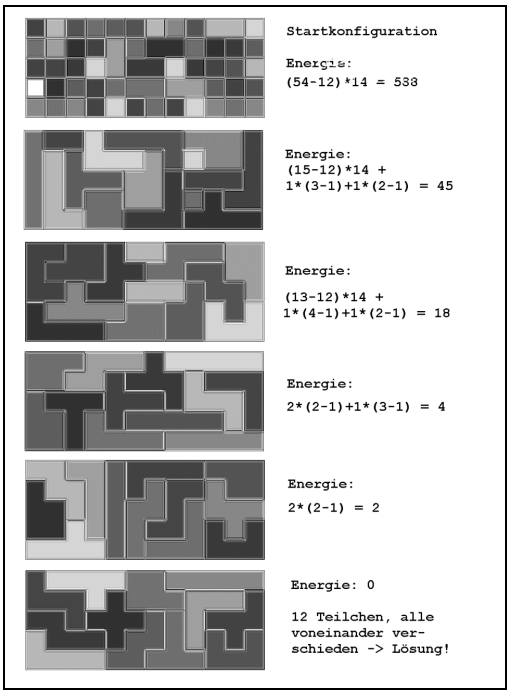
Ich möchte am Beispiel des Pentominopuzzles die Vorgehensweise von S kurz schildern:

Das Problem besteht darin, die 12 Pentominosteine lückenlos zu einem 5x12-Rechteck zusammenzusetzen. S modellierte zuerst die naheliegende „Schüttel- und Rüttelmethode“: Die 12 Teilchen befinden sich anfangs in der Gitterstruktur ausgerichtet im 5x12 Rechteck.

Dabei gibt es Einheitsquadrate die mehrfach oder gar nicht überdeckt sind. Die Tauschroutine wählt zufällig ein Teilchen, dreht oder spiegelt es zufällig und setzt es an einen zufälligen Ort. Die Energiefunktion kann aus der Anzahl nicht bedeckter oder der Anzahl mehrfach bedeckter Quadratfelder ermittelt werden. Die zweite Möglichkeit bietet eine quantitativ differenziertere Berwertung der neuen Anordnung. In beiden Fällen bedeutet $E(A)=0$, dass die Anordnung A optimal ist. S hat trotz diversen Änderungen an Tausch- und Energiefunktion und intensivem Experimentieren und Testen feststellen müssen, dass mit dieser „Schüttel- und Rüttelmethode“ nur in ganz seltenen Fällen eine Lösung generiert wird. Die schlechte Konvergenz dieses Verfahrens hat sie zudem „qualitativ“ begründet.

S gab sich nicht zufrieden. Statt die zwölf Puzzleteile durch Änderungen in Ort und Lage in eine optimale Anordnung zu transformieren, werden die Pentominosteine in 12 mal 5 Quadrate gleicher Farbe aufgelöst. Man startet mit einer beliebigen Anordnung der 60 Quadrate auf dem 12x5 Rechteck. Die Tauschmethode besteht allein darin, zwei zufällig ausgewählte Quadrate zu vertauschen. Die Energie einer Anordnung berechnet sich aus der Anzahl verschiedenfarbiger Gebiete und der Anzahl der Gebiete mit Fläche 5, welche einen bereits vorkommenden Pentominostein bilden. So berechnet sich die Energie der dritten Anordnung in nebenstehender Abbildung folgendermassen:

13 Gebiete	->+(13-12)*14	
1 Pentomino 4-fach	->	+1*(4-1)
1 Pentomino 2-fach	->	+1*(2-1)
Energie		18



Diese zweite Methode findet zuverlässig Lösungen für das 6x10-, 5x12-, 4x15- und das 3x20-Puzzle.

Bemerkungen

Die Maturaarbeit von S zeigt den praktischen Einsatz von Simulated Annealing an einer grossen Zahl verschiedenartiger Probleme auf. Die Qualität der Arbeit liegt deshalb in der Vielfalt und Breite der Behandlung des Themas. Dies liegt einerseits an der Faszination von S für die gefundene universelle Problemlösungsmethode, andererseits an den weit von der Mittelschulmathematik liegenden Werkzeugen, welche für eine quantitative Beurteilung dieser probabilistischen Verfahren nötig wären. Trotzdem hat S durch ihre Arbeit einige zum Teil neue Gebiete der Mathematik, wie Stochastik, Kombinatorik und Graphentheorie durch eigenständiges Erarbeiten kennen und anwenden gelernt. Die Hauptleistung der Arbeit liegt wohl in der Entwicklung von effizienten Algorithmen für überschaubare Teilprobleme (Beispiel: Wie bestimmt man die Anzahl Gebiete gleicher Farbe, in die eine Anordnung der

12x5 gleichfarbigen Quadrate zerfällt?). Das JAVA-Rahmenprogramm, der programmiertechnische Teil der Arbeit, sind im Teamwork mit dem Betreuer entstanden. Dadurch wurde S einerseits etwas entlastet, andererseits konnte S ihre Fähigkeit, neue Techniken und Ideen schnell zu verstehen und umzusetzen, unter Beweis stellen.

Fazit: Das Thema eignet sich für unterschiedliche Schülerprofile, wobei für alle Freude am zielgerichteten Programmieren und Interesse an theoretischen Aspekten der Informatik Voraussetzung ist:

- Ein konkretes Optimierungsproblem aus der Wirtschaft modellieren und mit Simulated Annealing lösen.
- Vergleich des probabilistischen mit dem exakten Algorithmus desselben Problems. Aufwandanalyse durch Zählen der Berechnungsschritte bei konkreten Programmläufen mit statistischer Auswertung.
- Simulation eines auf dem Extremalprinzip beruhenden physikalischen oder biologischen Phänomens (siehe auch Maturaarbeitsbericht im Bulletin Nr. 114).

Hinweise, Literatur:

- Informationen zum eingangs erwähnten Schnupperstudium: http://www.frauen.inf.ethz.ch/
- Applets zur Maturaarbeit von S: http://www.ateus.ch/fgmath/ProbAlg/ProbAlg.htm/
- Zeitschrift „Die Wurzel“ Doppelheft 3+4/98, Seiten 57-65: „Stundenplan leichtgemacht“
- Berthold Vöcking et altres, Taschenbuch der Algorithmen, Springer, Seiten 423-431 ISBN978-3-540-76393-2
- Juraj Hromkovic, Theoretische Informatik, Teubner, Seiten 268-273 ISBN 3-519-10332-X
- Juraj Hromkovic, Algorithmics for Hard Problems, Springer, Seiten 433-444 ISBN 3-540-44134-4