Anwendung evolutionärer Algorithmen
Übungsblatt 3

Karsten Weicker, Prof. Volker Claus, Abgabetermin: 12.05.99, 13:15 Uhr

Denkaufgabe: Eine in die Irre führende Funktion

1. Entwerfen Sie eine Funktion die auf dem Lösungsraum von 3 Bits definiert ist, die einen GA bezüglich der Schema-Theorie so in die Irre führen soll, daß der Algorithmus nicht auf dem Optimum 111 konvergiert, sondern zum Bitstring 000 hingeführt wird.

   D.h. die Fitneßfunktion ist so zu konstruieren, daß alle Schemata, die 000 enthalten (z.B. $00*$) im Durchschnitt besser bewertet werden als die anderen Schemata derselben Partition (im Beispiel $01*$, $10*$ und $11*$).

Denkaufgabe: Hybridisierung

1. Gegeben sei eine Funktion $f(x_1, x_2) = \sum_{i \in 1,2} (x_i - 1)^2$. Weiterhin sei eine lokale Suchprozedur gegeben, die auf jede Suchraumdimension zufällige Werte aus dem Intervall $[-0.05, 0.05]$ addiert und diese Änderungen nur dann akzeptiert, falls sie eine Verbesserung mit sich bringen.

   Berechnen Sie nun für die Suchraumpunkte $(0.7, 0.5)$, $(1.0, 0.9)$ und $(3.0, 2.0)$ die maximal mögliche Verbesserung nach 5 und 10 Lernschritten.

Kleine Programmieraufgabe: Sexuelle Selektion

Gegeben sei das Problem, daß in einem Bitstring der Länge 100 die Anzahl der 1en maximiert werden soll. Zusätzlich wird ein Bit in jedem Individuum eingeführt, welches angibt ob es sich um ein männliches (0) oder ein weibliches (1) Individuum handelt. Dieses Bit zählt nicht bei der Berechnung der Fitneß. Betrachten Sie den folgenden Algorithmus:

1. Initialisieren Sie eine Population mit weiblichen Individuen und eine zweite Population mit männlichen Individuen
2. Wählen Sie ein Weibchen fitneßproportional und ein Männchen fitneßproportional. Angenommen Weibchen bevorzugen Männchen mit mehr 0en: eine Funktion, die von der Anzahl der 0en des Männchens abhängt, gibt an, ob das Weibchen das Männchen akzeptiert. Falls sie nicht akzeptiert, wiederholen Sie die fitneßproportionale Auswahl eines Partners für das Weibchen so lange, bis ein passender Partner gefunden ist. Dann erzeugen Sie zwei Nachkommen mit 1-Punkt-Crossover. Plazieren Sie den weiblichen Nachkommen in einer Population für weibliche Nachkommen und das Männchen in einer Population für männliche Nachkommen.
3. Wiederholen Sie Schritt 2 so lange, bis die Populationen der Nachkommen genug Individuen enthalten. Dann ersetzen die Nachkommenspopulationen die Hauptpopulationen und Schritt 2 wird wieder durchgeführt mit neuen leeren Nachkommenspopulationen.

Aufgaben

1. Implementieren Sie einen solchen Algorithmus.
2. Was ist das Verhalten dieses GA?
3. Wie läßt sich das Verhalten erklären?
4. Experimentieren Sie mit verschiedenen Akzeptanzfunktionen für die Weibchen.

Programmieraufgabe: Hybridisierung

In diesen Übungen wird das Job-Shop-Scheduling-Problem betrachtet: In einem Fertigungsprozeß stehen n Maschinen zur Verfügung, auf denen m Aufträge ausgeführt werden sollen. Dabei ist für jeden Auftrag eine Ordnung vorgegeben, in welcher Reihenfolge welche Maschine für diesen Auftrag benötigt wird. Ebenso ist für jeden Auftrag und jede Maschine, eine Zeit angegeben, die für diese Teilauftrag benötigt wird.

Beispiel:

Reihenfolge						Zeit pro Masch.	M1	M2	M3	M4	M5
Job 1	1	2	3	4	5	Job 1	20	5	17	4	10
Job 2	4	2	1	3	5	Job 2	14	12	10	10	23
Job 3	4	5	3	2	1	Job 3	8	5	7	9	6
Job 4	4	3	2	1	5	Job 4	12	8	7	15	8
Job 5	3	1	5	4	2	Job 5	5	11	6	14	13

Gegeben ist nun ein evolutionärer Algorithmus, welcher die Gesamtzeit für einen Produktionsplan minimiert. Der Quelltext des Programms ist auf der WWW-Seite erhältlich.

Repräsentation: Permutationen der Zahlen $0, \ldots, (\text{Anzahl der Jobs}) * (\text{Anzahl der Maschinen}) - 1$

Dekodierung: Bei der Dekodierung werden die Zahlen $0, \ldots, (\text{Anzahl der Maschinen}) - 1$ als Teilaufträge des ersten Auftrags, $(\text{Anzahl der Maschinen}), \ldots, 2 * (\text{Anzahl der Maschinen}) - 1$ als Teilaufträge des zweiten Auftrags, etc. gewertet. Die Reihenfolge der Aufträge ergibt sich dabei aus der Tabelle des Problems.

Rekombination: ordnungserhaltender Crossover (vgl. erstes Übungsblatt)

Mutation: Vertauschung zweier Zahlen in der Permutation

Steady-state: Es wird nur eine neue Lösung pro Generation erzeugt. Diese ersetzt die schlechteste Lösung in der Population.

Aufgaben

1. Testen Sie das vorhandene Programm auf verschiedenen Datensätzen (WWW-Seite). Wenn möglich, veranschaulichen Sie eine Lösung für eine kleine Probleminstanz als GANTT-Diagramm.
2. Überlegen Sie sich verschiedene Möglichkeiten, wie der evolutionäre Algorithmus durch Hybridisieren verbessert werden kann. Beachten Sie dabei die folgenden Möglichkeiten.
   (a) Kann lokale Suche mit eingebunden werden?
   (b) Kann eine Heuristik oder deterministische Strategie zur Verbesserung einer Lösung miteinbezogen werden?

Stellen Sie die möglichen Ansätze zur Hybridisierung kurz dar.

3. Implementieren Sie wenigstens einen hybriden Algorithmus.

4. Was konnte durch diese Strategie erreicht werden. Diskutieren Sie kurz die Resultate in Bezug auf die Qualität der Lösung und die Laufzeit (Anzahl der Fitneßberechnungen). Die Lösung einer kleinen Probleminstanz kann wieder als GANTT-Diagramm veranschaulicht werden.