Matthias Jauernig
06INM

Lehrveranstaltung „Evolutionäre Algorithmen“
Dozent: Prof. Weicker

Projektvorstellung „Zeitreihen-Modellierung mit Gene Expression Programming“

1 „Gene Expression Programming“ kurz erklärt

Das *Gene Expression Programming* (GEP) wurde 2001 von Candida Ferreira vorgestellt[1]. Die ursprüngliche Idee war dabei ähnlich wie beim genetischen Programmieren ausführbare Programme zu evolvieren. Es handelt sich bei GEP um eine Weiterentwicklung der Ideen *genetischer Algorithmen* (GAs) und eben des *genetischen Programmierens* (GP), wobei ebenso wie bei diesen Algorithmen Populationen von Individuen benutzt werden, die anhand ihrer Fitness selektiert und genetischen Operatoren unterworfen werden, um genetische Variation zu erreichen. Der grundsätzliche Unterschied bzw. die neue Idee hinter GEP besteht im Aufbau der Individuen. Bei GAs stellen diese symbolische Strings fester Länge dar, bei GP handelt es sich um nicht-lineare Bäume (*parse trees*) unterschiedlicher Größe und Aufbaus. Im GEP unterscheidet man zwischen *Genotypen*, welche als *Chromosomen* Strings fester Länge darstellen und *Phänotypen*, die eindeutig aus Genotypen generiert werden können und die in Form von Ausdrucksbäumen repräsentiert werden. Durch Inorder-Traversierung eines solchen Baumes lässt sich beispielsweise Code für ausführbare Programme generieren.

1.1 Genotyp und Phänotyp

Der große Fortschritt des GEP besteht in der Einführung von Chromosomen als Genotypen, die in der Lage sind trotz ihrer festen Länge variable Ausdrucksbäume (*expression trees*, ETs) zu generieren. Im Gegensatz zum GP werden genetische Operatoren direkt auf den Chromosomen und nicht auf den ETs ausgeführt und erst danach in Ausdrucksbäume überführt. Wie noch zu sehen sein wird, bedeutet dies eine starke Erhöhung der Flexibilität, wenn es um die Frage der syntaktischen Korrektheit von Programmen geht.
Die Struktur eines Genotyp erklärt sich wie folgt: ein Genotyp (im Weiteren auch als *Chromosom* bezeichnet) ist generell eine Aneinanderreihung einzelner Zeichen. Bei diesen Zeichen unterscheidet man weiterhin zwischen *Funktionssymbolen* (FS) und *Terminalsymbolen* (TS). Ein Chromosom kann dabei logisch gesehen in einzelne *Gene* unterteilt werden. Alle Gene eines Chromosoms haben dabei dieselbe Länge (also die gleiche Anzahl an Zeichen). Ein Gen besteht wiederum aus einem *Kopf* und einem *Rest*. Kopf und Rest beinhalten dann die einzelnen Zeichen. Kopflänge h und Restlänge t stehen dabei immer in folgendem Verhältnis zueinander:

$$t = h \cdot (n - 1) + 1$$

n stellt hierbei die höchste Arität eines FS dar. Weiterhin gilt, dass im Gen-Kopf FS und TS beliebig vorkommen können, der Gen-Rest jedoch nur aus TS besteht. Zusammen mit dem Verhältnis $t : h$ sichert dies die Überführbarkeit des Chromosoms in einen Ausdrucksbaum und die Besetztheit der Baumblätter mit TS.
Ein Phänotyp ist weniger kompliziert aufgebaut. Es handelt sich dabei lediglich um einen Baum allgemeiner Struktur, der in seinen inneren Knoten FS und in seinen Blättern TS besitzt. Durch Inorder-Traversierung des Baumes erhält man den Ausdruck, welchen dieser repräsentiert.

[1] Candida Ferreira 2001, „Gene Expression Programming: A New Adaptive Algorithm for Solving Problems“, http://www.gene-expression-programming.com

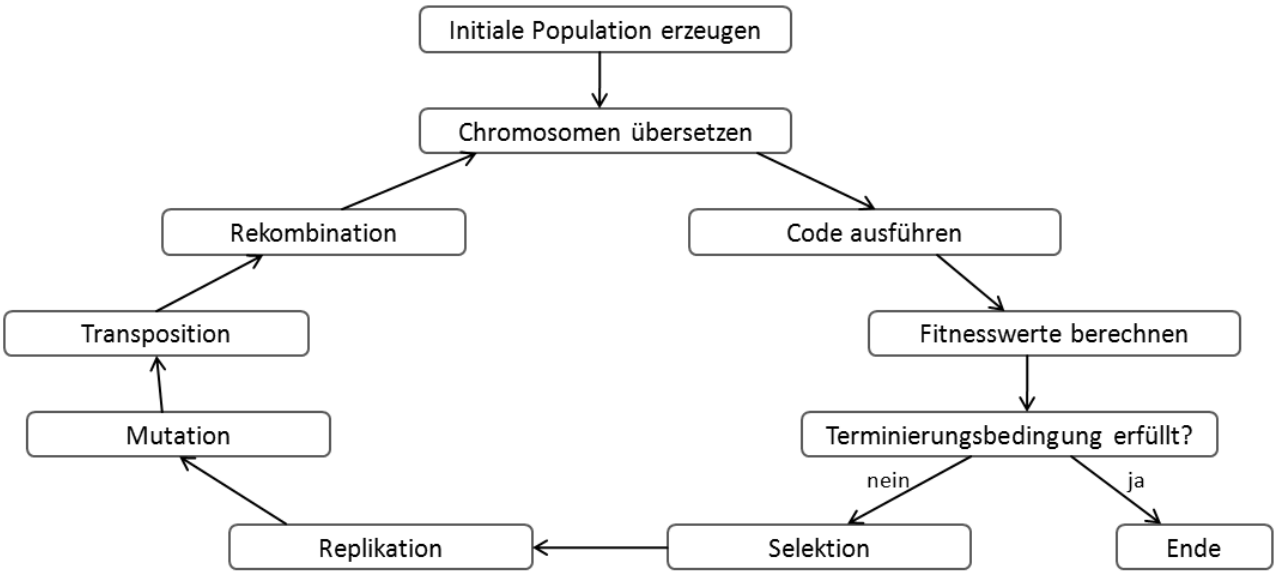
Jeder Genotyp lässt sich nun eindeutig in einen Phänotyp überführen. Dabei wird jedes Gen eines Chromosoms in einen eigenen Ausdrucksbaum (ET) umgewandelt (einzelne Ausdrucksbäume eines Phänotyps (als Liste von Bäumen) werden dann i.d.R. wieder miteinander verknüpft). Dies geschieht durch sequentielles zeichenweises Abarbeiten des jeweiligen Gens, wodurch man bei Entnahme eines FS offene Stellen entsprechend der Arität des FS erhält, die durch nachfolgende Zeichen zu besetzen sind. Diese Zeichen stellen die Söhne eines durch einen Knoten repräsentierten FS im Baum dar. Es ergibt sich ein ebenenweiser Aufbau des ET, bis alle offenen Stellen besetzt sind. Ist dies der Fall, so terminiert der Algorithmus. Das bedeutet gleichermaßen, dass im Extremfall alle Symbole des Gens im Baum Verwendung finden, im anderen Extrem der Baum hingegen aus nur einem Knoten besteht, wenn das Gen mit einem TS beginnt und keine Söhne besetzt werden müssen. Den Teil eines Gens, welcher im Baum repräsentiert wird, nennt man auch *kodierenden Teil* des Gens. Der nicht-kodierende Genteil ist zunächst für das aktuelle Individuum nutzlos, kann jedoch unter Anwendung genetischer Operatoren für Kind-Generationen aktiviert werden. Es ergibt sich dadurch ein flexibles Zusammenspiel zwischen Genotyp und Phänotyp, da Gene fester Länge Ausdrucksbäume variabler Größe und Struktur erzeugen können. Zudem umgeht man im Gegensatz zu GP bei der Anwendung genetischer Operatoren die Probleme der syntaktischen Korrektheit zu einem großen Teil, wenn die Struktur des Gens eingehalten wird.

1.2 Genetische Operatoren

C. Ferreira führt in Ihren ursprünglichen Ausarbeitungen eine Reihe genetischer Operatoren an, die im GEP Verwendung finden. Zunächst gibt es die *Selektion* zu nennen, welche auf einem fitnessproportionalen Roullette-Rad-Schema basiert, dessen Gewinner repliziert werden. Die *Mutation* wird als Hauptfaktor der Diversitätserhaltung genannt, wobei im Gen-Kopf beliebig mutiert werden darf, im Gen-Rest hingegen nur TS ausgetauscht werden können, um die alles entscheidende Gen-Struktur zu erhalten. Mit der *Transposition* werden Genomteile innerhalb eines Chromosoms verschoben, bei der *Rekombination* findet dieser Prozess über mehrere Chromosomen hinweg statt. Die einzige Einschränkung ist die Erhaltung der Gen-Struktur, um die Überführbarkeit in den Phänotyp zu bewahren.

1.3 Ablaufdiagramm

Die allgemene Struktur des GEP-Algorithmus' ergibt sich wie folgt:



1.4 Anwendungsgebiete

Die Webseite zu GEP gibt Beispiele für Anwendungsgebiete von GEP und Probleme, die damit gelöst wurden bzw. Gegenstand aktueller Untersuchungen sind. Dazu zählen u. a.:

- Symbolische Regression
- Zeitreihen-Voraussage
- Block stacking
- Klassifikationsprobleme
- Logik-Synthese
- Parameter-Optimierung
- Design Künstlicher Neuraler Netze
- Scheduling
- Zelluläre Automaten

2 Projektziel

Ich möchte in meinem Projekt die Modellierung von Zeitreihen mithilfe des Gene Expression Programming behandeln. Ausgehend von einem allgemeinen Begriff der Zeitreihe sollen dabei ein paar wenige Beispiele zur Veranschaulichung und als Testmaterial dienen. Es steht dabei im Vordergrund, durch die Evolution auf einem Satz von Basis-Operatoren und Symbolen eine Funktion zu finden, welche die zugrunde gelegte Zeitreihe approximiert. Weiterhin werden die verfügbaren Daten einer Zeitreihe in eine Trainingsmenge sowie eine Testmenge unterteilt und es wird geprüft, inwieweit die evolvierte Funktion in der Lage ist, den Daten dieser Mengen zu entsprechen. Weiterhin ist die Auswahl genetischer Operatoren und vor allem die Parameterwahl ein wichtiges Problem, welches empirisch gelöst werden soll, wobei auch Wert auf die Begründung der Lösung in vergleichender Form gelegt wird.

3 Teilaufgaben

Im theoretischen Teil sind folgende Aufgaben zu lösen:

1. Problembeschreibung
2. Kurzbeschreibung von GEP und den genetischen Operatoren
3. Realisierung und Parameter
4. Beschreibung der erlangten Ergebnisse
5. Vergleich mit anderen Methoden

Der Praxisteil beinhaltet die folgenden Aufgaben:

1. Erstellung einer allgemeinen Programmstruktur
2. Umsetzung der Datenstrukturen Genotyp und Phänotyp
3. Umsetzung genetischer Operatoren
4. Implementierung der Problemdomäne
5. Finden geeigneter Parameter zur Lösung des Problems