Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 86 (1995)

Heft: 25

Artikel: Optimieren nach dem Vorbild der Natur : evolutionäre Algorithmen

Autor: Blickle, Tobias

DOI: https://doi.org/10.5169/seals-902520

Nutzungsbedingungen

Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften auf E-Periodica. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Das Veröffentlichen von Bildern in Print- und Online-Publikationen sowie auf Social Media-Kanälen oder Webseiten ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Mehr erfahren

Conditions d'utilisation

L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. La reproduction d'images dans des publications imprimées ou en ligne ainsi que sur des canaux de médias sociaux ou des sites web n'est autorisée qu'avec l'accord préalable des détenteurs des droits. En savoir plus

Terms of use

The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. Publishing images in print and online publications, as well as on social media channels or websites, is only permitted with the prior consent of the rights holders. Find out more

Download PDF: 10.12.2025

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

Mit dem Streben, die Regeln der Natur zu verstehen, war auch schon immer der Wunsch da, ihre Methoden nutzbringend für menschliche Kreationen anzuwenden. Seit einigen Jahren ist ein grosses Interesse an derartigen Verfahren festzustellen, vor allem weil die klassischen Verfahren für viele Probleme noch keine befriedigende Lösung bieten. Neben den neuronalen Netzen, die in den Illustrierten für Schlagzeilen gesorgt haben, gehören die evolutionären Algorithmen wohl zu den interessantesten derzeitigen Forschungsgebieten.

Optimieren nach dem Vorbild der Natur

Evolutionäre Algorithmen

■ Tobias Blickle

Evolutionäre Algorithmen sind eine Klasse von probabilistischen Optimierungsverfahren, welche «Ideen» der natürlichen Evolution nutzen. Wegen ihrer Universalität und Robustheit eignen sie sich besonders für komplexe Optimierungsprobleme, bei denen klassische Optimierungsverfahren versagen. Der Artikel stellt die Arbeitsweise der evolutionären Algorithmen am Beispiel des Parkierens eines LKW vor, diskutiert die Vor- und Nachteile des Verfahrens und verweist auf einige interessante Anwendungen.

Optimierungsprobleme zählen schon fast zum täglichen Brot des Ingenieurs. In vielen Bereichen – beim Routen einer Platine, bei der Ablaufplanung von Prozessen, der Bestimmung von Parametern einer Regelung, der Modellierung chemischer und physikalischer Vorgänge oder in der Bildverarbeitung – trifft man auf Probleme, die sich mit der Frage umschreiben lassen: Wie sieht die Lösung meines Problems aus, die meine Kostenfunktion minimiert (oder maximiert) und gleichzeitig auch meine Nebenbedingungen einhält? Die meisten dieser Probleme sind NP-hart, das heisst, sie gehören in die Klasse der schwierigsten Probleme, die die Informatik kennt. Die Suche nach exakten Lösungen für NP-harte Probleme treibt die erforderliche Rechenzeit jedes Supercomputers ins «Unendliche», da die Laufzeit exponentiell mit der Eingabegrösse ansteigt. Deshalb wendet man für grosse Optimierungsprobleme sogenannte heuristische Algorithmen an. Diese garantieren zwar nicht, dass das globale Optimum gefunden wird, finden dafür aber in vertretbarer Zeit eine «brauchbare» Lösung. In vielen Fällen genügt es nämlich, überhaupt eine Lösung zu finden; es muss nicht immer die denkbar beste sein. Bei der Verdrahtung einer Platine beispielsweise ist es wichtiger, ein Layout zu finden, das alle Pins richtig miteinander verbindet, als jenes Layout, in dem die Verbindungen auch noch optimal kurz sind. Für viele Optimierungsprobleme gibt es speziell angepasste Heuristiken. Aber es gibt auch Optimierungsverfahren wie das «Simulated Annealing» (Simuliertes Erstarren) [1], die für verschiedenste Aufgaben eingesetzt werden können, weil sie relativ unabhängig von der Problemstellung sind. Eine derartige Ungebundenheit und die Aussicht auf zahlreiche potentielle Anwendungen zeichnen auch die evolutionären Algorithmen aus.

Arbeitsweise evolutionärer Algorithmen

Die evolutionären Algorithmen finden inzwischen in vielen Bereichen Anwendung, und einige davon sind in einem separaten Kasten vermerkt. Hier soll ihre Arbeitsweise am Problem des Parkierens eines LKW erläutert werden [3]. Diese

Adresse des Autors:
Tobias Blickle, Dipl.-Ing., Institut für Technische Informatik und Kommunikationsnetze, ETH Zentrum, 8092 Zürich.

Anwendungsbeispiele evolutionärer Algorithmen

Im folgenden sollen einige Anwendungsbeispiele evolutionärer Algorithmen aufgeführt werden. Die Liste erhebt keinen Anspruch auf Vollständigkeit; sie soll nur den grossen Einsatzbereich dieser Optimierungsmethode dokumentieren. Viele Beispiele sind in [6] gesammelt; einen guten Überblick über die momentane Forschung und aktuelle Anwendungen geben auch die Tagungsbände der beiden wichtigsten Konferenzen über evolutionäre Algorithmen [7; 8].

Operations Research:
- Optimale Ablaufplanung von Prozessen
- Problem des Handlungsreisenden
- Partitionierungsprobleme

Robotik:
- Optimale Trajektorien für Roboterarme
- Wegeplanung von Robotern

Künstliche Intelligenz:
- Steuerung autonomer Fahrzeuge
- Optimierung der Gewichte neuronaler Netze
- Optimierung der Topologie neuronaler Netze

Telekommunikation:
- Optimierung der Netzwerkstruktur
- Optimierung der Nachrichtenvermittlung (Message Routing)

Mikroelektronik:
- Parameteroptimierung digitaler Filter
- Partitionierung digitaler Schaltungen

Allgemeine Optimierungen:
- Optimierung einer Pipeline-Steuerung
- Optimierung von physikalisch/chemischen Modellen
- Optimierung von Parametern einer Regelung
- Optimierung der Regelbasis von Fuzzy-Reglern

Aufgabe ist sehr anschaulich und dient deshalb oft als Demonstrationsbeispiel in der Regelungstechnik. So existieren zum Beispiel auch Lösungen zu diesem Problem, bei denen die Regelung durch ein neuronales Netz erfolgt. Die Aufgabenstellung lautet dabei wie folgt: Ein LKW steht an einer beliebigen Position auf einem Hof und soll innerhalb einer gewissen Zeitspanne rückwärts an eine Laderampe (Position 0,0) eingeparkt werden (Bild 1). Der LKW fährt dabei mit konstanter Geschwindigkeit rückwärts. Gesucht ist nun ein Steuerprogramm, das aus der momentanen Position des LKW (x, y) relativ zur Laderampe, dem Winkel zwischen Führerhaus und Anhänger (diff) und dem Winkel des Anhängers bezogen auf die X-Achse (tang) zu jedem Zeitpunkt den richtigen Radeinschlag bestimmt und so den LKW einparkt. Unser Optimierungsproblem lautet also: Wie muss die Regelungsfunktion aussehen, damit der Fehler beim Einparken aus einer beliebigen Anfangsposition minimal ist, unter der Bedingung, dass eine vorgegebene Zeitbeschränkung nicht verletzt wird? Es wird also ein funktionaler Zusammenhang **F** zwischen den Eingabegrössen (x, y, tang, diff) und der Ausgabegrösse u gesucht, das heisst

$$u = F(x, y, tang, diff) \qquad (1)$$

Die Einzelheiten zu dieser Problemstellung sind im Kasten «Einparken eines LKW» beschrieben.

Mittels evolutionärer Algorithmen wird dieses Problem auf folgende Weise gelöst:

Grundlage ist eine Menge (Population) von möglichen Lösungen des Optimierungsproblems (Individuen). Diese Individuen können je nach Aufgabe in ganz unterschiedlicher Form dargestellt sein. Für die Darstellung von funktionalen Zusammenhängen, wie bei dem hier betrachteten Einparkproblem, eignet sich am besten die Baumstruktur. Andere bekannte Arten und Weisen, die Individuen darzustellen, sind Bitstrings (Zeichenketten konstanter Länge aus 0 und 1) und Vektoren von reellen Zahlen (für die Optimierung von Parametern).

Zu Anfang wird die Population mit zufällig erzeugten Individuen belebt (Bild 2), das heisst, es werden irgendwelche Steuerprogramme aus den gewählten Funktionen und Argumenten des LKW-Beispiels gewürfelt. In diesem Fall wird also «bei Null» angefangen und kein Wissen über die Lösung eingebracht, ausser durch die Wahl des «Befehlssatzes» (also welche Funktionen und Argumente zulässig sind). Die meisten der Individuen werden das Problem schlecht oder gar nicht bewältigen, aber sicherlich werden einige «besser» sein als andere. Was «besser» heisst, wird dabei durch die sogenannte Fitnessfunktion bestimmt, die jedem Individuum einen Zahlenwert zuordnet, der die Güte des Individuums bezüglich des Optimierungsproblems angibt. Im Beispiel gibt die Entfernung des LKW von der Laderampe nach dem Einparkvorgang die Fitness eines Steuerprogramms (Individuums) an.

Ziel des evolutionären Algorithmus ist nun, ein Individuum mit optimaler Fitness zu finden. Um dies zu erreichen, werden durch Selektion und Rekombination immer wieder neue Populationen von potentiellen Lösungen erzeugt, bis man mit der Qualität der Lösung des besten Individuums zufrieden ist und das Abbruchkriterium erfüllt ist. Durch das Zusammenspiel von Selektion und Reproduktion erhofft man sich eine Verbesserung der Fitnesswerte (und schliesslich die optimale Lösung).

Die Selektion soll die mittlere Fitness der Population verbessern, indem sie durch ein geeignetes Verfahren gute Individuen (im Sinne der Fitnessfunktion) häufiger für die nächste Population auswählt als schlechte. Die Art und Struktur der Individuen wird dabei nicht verändert, es werden nur identische Kopien erzeugt. Dadurch werden mehr «Testpunkte» (Individuen) an Stellen im Lösungsraum plaziert, die einen relativ guten Fitnesswert haben. Die Selektion führt somit zu einer Konzentration der Population auf untersuchte und gute Gebiete im Lösungsraum. Die Selektionsmethoden gehen im allgemeinen probabilistisch vor; übliche Selektionsmethoden sind zum Beispiel:

- Proportionale Selektion: Die Selektionswahrscheinlichkeit eines Individuums ist proportional zu seiner Fitness.

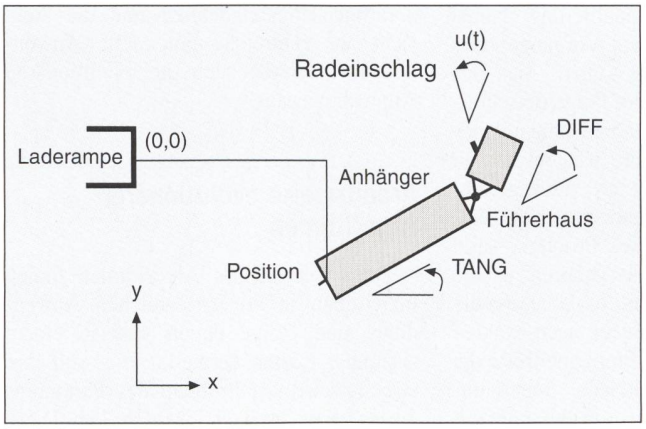

Bild 1 Szenario zum Rückwärtseinparken eines LKW

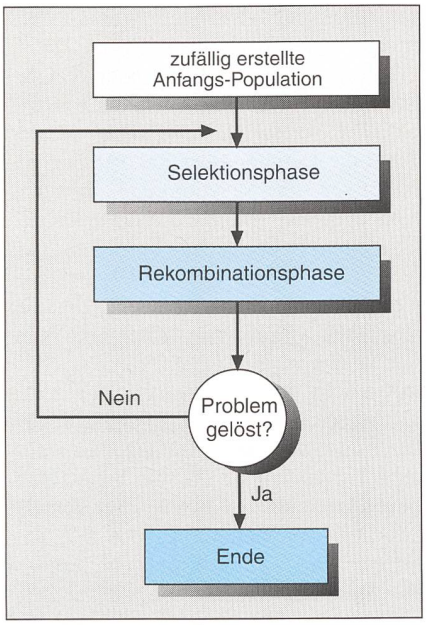

Bild 2 Ablaufplan eines evolutionären Algorithmus

- Truncation-Selektion: Nur ein bestimmter Prozentsatz der besten Individuen der Population überlebt – was tatsächlich eine Art des «survival of the fittest» bedeutet.
- Tournament-Selektion: Jedes Individuum der Nachfolgepopulation geht aus einem Wettkampf zwischen einer bestimmten Anzahl von zufällig ausgewählten Individuen hervor.

Die Aufgabe der Rekombination ist, «inhaltlich» neue Individuen zu erzeugen, um bessere Lösungen zu finden. Auch hier spielt der Zufall eine grosse Rolle. Möglich sind sowohl ungeschlechtliche als auch geschlechtliche Rekombinationen. Bei der ungeschlechtlichen wird durch Mutation ein Teil der Information (z. B. Gleichungsstruktur des Individuums bei Bäumen) zufällig geändert. Bei der geschlechtlichen Rekombination – dem Kreuzen zweier (oder mehrerer) Individuen – wird dagegen ein Teil der Information zwischen den Individuen getauscht. Die Veränderung der Struktur der Individuen bedeutet, dass neue «Punkte» im Lösungsraum durch Individuen besetzt und somit neue Gebiete im Lösungsraum «erforscht» werden. Normalerweise ändert man nicht die ganze Population, sondern nur einen gewissen Prozentsatz. Verschiedene Rekombinationsoperatoren (siehe weiter unten) können auch gleichzeitig angewendet werden, und gewöhnlich ist jedem Rekombinationsoperator eine Ausführungswahrscheinlichkeit zugeordnet. In den Bildern 5 bis 9 sind verschiedene Rekombinationsarten dargestellt.

Bei unserer Aufgabe, den LKW optimal zu parkieren, sind die Individuen als Bäume dargestellt, und so lässt sich das Kreuzen durch Austauschen zufälliger Teilbäume zwischen den Individuen realisieren (Bild 7).

Das Zusammenspiel dieser beiden Hauptakteure – Selektion und Rekombination – führt mit der Zeit (Generationen) zu immer «besseren» Individuen. Die Lösung des LKW-Problems wurde mit einer Populationsgrösse von 500 Individuen versucht, sie gelang nach 11 Generationen. Die gefundene Regelfunktion lautet:

u= PLUS[MINUS[PLUS[MINUS
[PLUS[diff,diff],IFLTZ[x,x,tang]],
IFLTZ[MUL[tang,diff],MINUS[diff,tang],y]],
MINUS[MUL[tang,x],y]],IFLTZ[MUL[tang,diff],
MINUS[diff,tang],PLUS[tang,y]]] (2)

wobei u das Ergebnis, das heisst der Radeinschlag ist, der ausgeführt wird, wenn der LKW an der Stelle (x, y) auf dem Hof steht und die Winkel tang bzw. diff sind. Bild 3 zeigt die Lösung als Baumstruktur. Diese Funktion ist natürlich nicht «die» Lösung, denn es gibt sehr viele Lösungen für das Problem, und lässt man das Problem mehrmals durch einen evolutionären Algorithmus lösen, wird man sehr wahrscheinlich jedesmal eine andere Lösung erhalten, oft genug sogar gar keine.

Vom «gemeinen Nutzen» evolutionärer Algorithmen

Bei den evolutionären Algorithmen handelt es sich also um eine Art «Black-Box»-Optimierung: das Grundgerüst der Selektion und Rekombination bleibt konstant und ist unabhängig von der Optimierungsaufgabe. Für eine bestimmte Aufgabe müssen im wesentlichen zwei Dinge angepasst werden: die Fitnessfunktion und die Repräsentation der Lösung.

Die Fitnessfunktion ist der einzige Mechanismus, über den der evolutionäre Algorithmus Informationen über die Qualität seiner momentanen Population erhält. Dies hat den Vorteil, dass man auch sehr komplexe (unstetige, nicht differenzierbare oder abschnittsweise definierte) Funktionen optimieren kann. Einzige Voraussetzung ist, dass man zu jedem möglichen Individuum (Punkt im Lösungsraum) den Fitnesswert ermitteln kann. Das bedeutet aber auch, dass der sensitive Punkt dieses Optimierungsverfahrens in

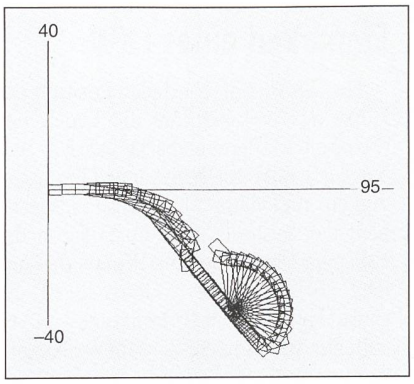

Bild 4 Einparkvorgang mit der Lösung aus Bild 3

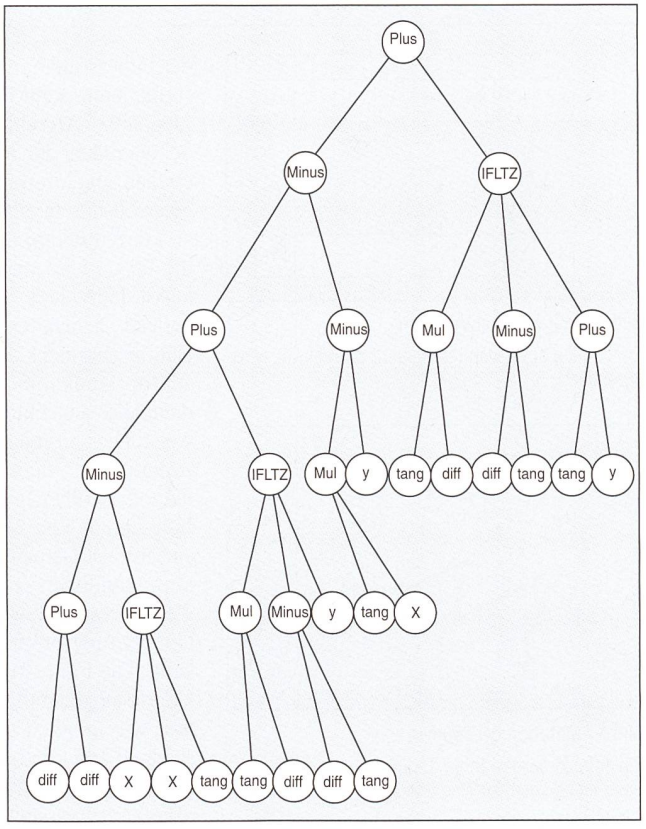

Bild 3 Eine Lösung des Einparkproblems

Einparken eines LKW

Der LKW soll von jeder Position des Hofes aus rückwärts an die Laderampe (Position (0,0)) eingeparkt werden, wobei der LKW mit konstanter Geschwindigkeit rückwärts fährt. Zustandsvariablen des Systems sind die Position des Anhängers (x, y) und die Winkel diff und tang (Bild 2). Die Stellgrösse ist der Einschlagwinkel des Lenkrades u. Es soll ein funktionaler Zusammenhang **F** bestimmt werden, so dass zu jeder Position der richtige Lenkradeinschlag u erfolgt. Zum Bestimmen der Qualität der Funktion **F** (also der Fitness des Individuums) wird der LKW von acht verschiedenen Positionen gestartet und die Summe der Entfernungen nach dem Einparken aller acht Testfälle gebildet. Das Problem gilt als gelöst, wenn diese Summe unter einer vorgegebenen Schranke liegt und die vorgegebene maximale Zeit für den Einparkvorgang nicht überschritten wurde.

Folgende Operatoren wurden gewählt, aus denen die Funktion aufgebaut werden kann:

PLUS(a,b):	Addition a+b
MINUS(a,b):	Subtraktion a–b
DIV(a,b):	(geschützte) Division a/b
MUL(a,b):	Multiplikation a*b
ATG(a,b):	ArcTan(b/a)
IFLTZ(a,b,c):	Wenn (a < 0) gib b zurück, sonst c

Terminale (Variablen und Konstanten) des Baumes sind:

x	momentane x-Position des LKW-Anhängers
y	momentane y-Position des LKW-Anhängers
diff	momentaner Winkel zwischen Führerhaus und Anhänger
tang	momentaner Winkel zwischen Anhänger und x-Achse
RR	reelle Zufallszahl

Als Programmparameter werden vorgegeben:
- Populationsgrösse: 500
- Tournament-Selektion: mit Gruppengrösse 10
- Kreuzungswahrscheinlichkeit 90%
- keine Mutation

Die Lösung nach 11 Generationen (es wurden 5401 Individuen erzeugt) zeigt Bild 3. Wie man sieht, besitzt die Lösung links beginnend die algebraische Struktur von Formel 2. Eine aus der Lösung von Bild 3 resultierende Einparksequenz zeigt Bild 4.

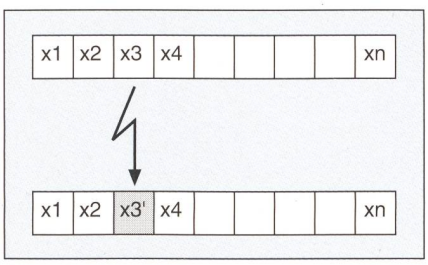

Bild 5 Mutation bei Vektoren

Ein zufällig ausgewähltes Element des Vektors wird verändert.

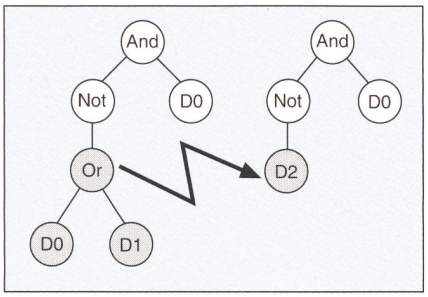

Bild 6 Mutation bei Bäumen

Ein zufällig ausgewählter Teilbaum wird durch einen beliebigen, zufällig erzeugten Teilbaum ersetzt.

der Fitnessfunktion liegt; diese «richtig» zu formulieren, stellt bei der Anwendung evolutionärer Algorithmen oft die grösste Schwierigkeit dar. Meist ist es nämlich nicht einfach, alle Informationen über die Qualität einer Lösung in einem einzigen Zahlenwert zu konzentrieren. Beispielsweise stellt sich die Frage, wie «ungültige» Individuen, das heisst Individuen, die eine Nebenbindung verletzt haben, bewertet werden sollen. Häufig geschieht das durch eine Art «Bestrafungsfunktion», die ungültigen Individuen die schlechtestmögliche Fitness zuweist. Dies birgt einige Nachteile, denn der evolutionäre Algorithmus hat keine Information darüber, wie weit das ungültige Individuum von einer zulässigen Lösung entfernt ist. Dementsprechend werden auch alle ungültigen Lösungen untereinander gleich behandelt, was unter Umständen den Optimierungsprozess stark verlangsamen kann. Die Fitnessfunktion sollte also – das kann man generell sagen – soviel Information wie möglich enthalten und möglichst graduell und kontinuierlich aufgebaut sein.

Der zweite wesentliche Punkt bei der «Problemanpassung» besteht in der Wahl der Codierung eines Individuums (Bitstring, Vektor, Baum oder sonstige Datenstruktur). Bereits die Wahl einer geeigneten Datenstruktur kann den Suchprozess erheblich vereinfachen. Automatisch geht damit die Definition der Rekombinationsoperatoren für Mutation und Kreuzung einher. Aus informationstheoretischen Überlegungen dachte man lange Zeit, dass die besten Ergebnisse erzielt würden, wenn jedes Problem auf einen Bitstring abgebildet wird. Damit werden aber unter Umständen recht komplizierte Zwischenstufen nötig, um die Information zwischen Codierungsraum (Bitstrings) und Lösungsraum (also der Problemstellung) umzuformen. Ausserdem ist jetzt die Fitnessfunktion tatsächlich die einzige Schnittstelle zwischen Optimierungsmethode und Problemstellung. Oft lässt sich ein Problem aber besonders elegant und natürlich mit einer anderen Datenstruktur darstellen, wie beispielsweise die Regelungsfunktion für den zu parkenden LKW durch einen Baum. «Problemwissen» kann so nutzbringend in den Optimierungprozess eingebaut werden. Viele erfolgreiche Anwendungen evolutionärer Algorithmen beruhen auf diesem Prinzip [4; 5]. Häufig kann man den Rekombinationsoperator so gestalten, dass nur zulässige Lösungen erzeugt werden. Die Schwierigkeiten der diskutierten Bestrafungsfunktionen können auf diese Weise umgangen werden.

Hat man diese beiden Schritte für ein spezifisches Optimierungsproblem gelöst, kann das Verfahren direkt angewendet werden.

Schattenseiten

Die Vorteile der evolutionären Algorithmen haben leider auch ihren Preis. Es handelt sich um ein probabilistisches Verfahren, und man kann im allgemeinen keine Angaben über die Konvergenzgeschwindigkeit des Verfahrens (Komplexität des evolutionären Algorithmus) und über die Qualität der Lösung machen. Zwar kann man zeigen, dass bei beliebig langer Rechenzeit und bestimmten Selektionsmethoden das Optimum gefunden werden kann, aber diese Erkenntnis ist eher von akademischem Interesse.

Ein weiterer Nachteil ist die benötigte hohe Rechenleistung und der grosse Speicherbedarf. Da man eine Population von Individuen hat, muss man die ganze Population speichern, und je nach Problemstellung kann ein einzelnes Individuum schon einige Kilobyte Speicherplatz belegen. Eine typische Population kann 1000 und mehr Individuen haben, so dass man schnell einige Megabyte Speicherplatz be-

Künstliche Welten

Eine besonders spektakuläre Anwendung evolutionärer Algorithmen stammt von Karl Sims, der bei Thinking Machines arbeitet. Er hat mit genetischen Algorithmen künstliche Kreaturen erzeugt, die schwimmen oder springen oder sich auf einen Lichtpunkt zubewegen können. Die Kreaturen sind aus quaderähnlichen Elementen aufgebaut, die über «Gelenke» miteinander verbunden und mit «Berührungssensoren» ausgestattet sind. Die Sensoren und Aktoren werden durch ein künstliches neuronales Netz gesteuert. Der Bauplan für die Kreatur und die Topologie des neuronalen Netzes wird in einer graphenähnlichen Datenstruktur gespeichert, die von einem genetischen Algorithmus kontrolliert wird. Die Kreaturen «leben» in einer dreidimensionalen Welt, in der die «üblichen» physikalischen Gesetze gelten. Durch Optimierung der Kreaturen bezüglich der Schwimm- bzw. Springeigenschaften konnten tatsächlich schwimmende bzw. springende Kreaturen «gezüchtet» werden. Einige der Kreaturen sahen dabei wirklichen Lebewesen sehr ähnlich, während andere völlig neue «Lebensformen» darstellten.

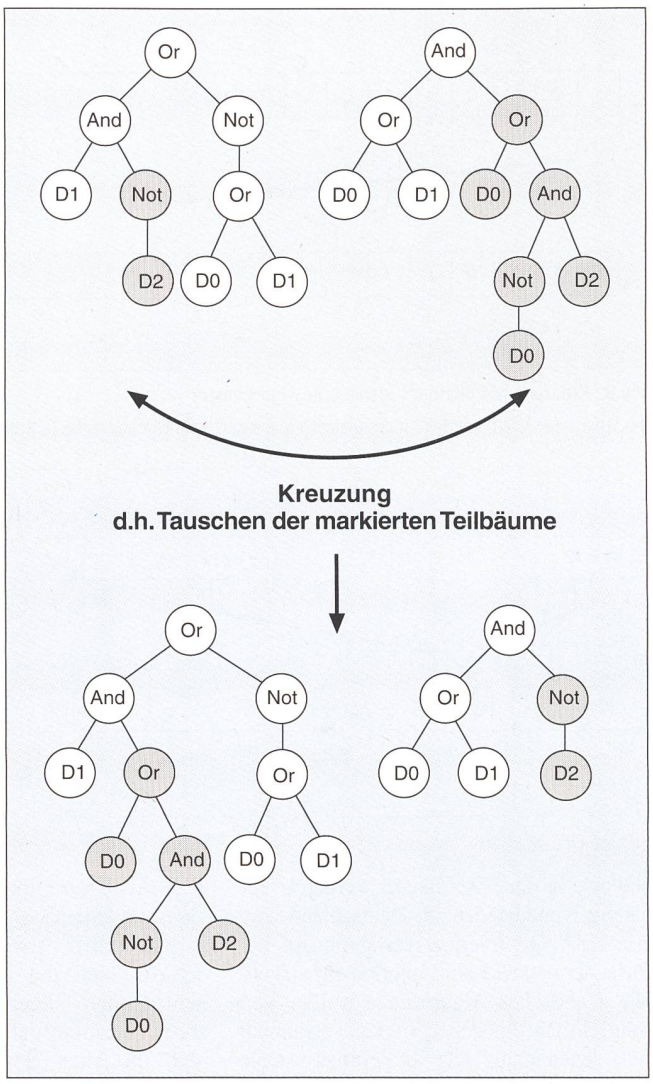

Bild 7 Kreuzung mit Bäumen (genetisches Programmieren)
Je ein Teilbaum wird zufällig ausgewählt, und dann werden die Teilbäume zwischen den Eltern getauscht, um die Kinder zu erhalten.

legt. Ausserdem müssen in jeder Generation die Fitnesswerte von praktisch allen Individuen neu berechnet werden, was sehr aufwendig sein kann. Im LKW-Beispiel müssen für jedes Individuum acht vollständige Einparkvorgänge simuliert werden, da von acht unterschiedlichen Positionen auf dem Hof gestartet wird, und jede Simulation besteht aus bis zu 3000 Simulationsschritten.

Diesem Problem kann man mit einer Parallelisierung des Algorithmus allerdings teilweise begegnen. Das geschieht folgendermassen: Man spaltet die Population in mehrere Teilpopulationen auf und ordnet jeder Teilpopulation einen eigenen Prozessor zu. Bei lokaler Selektion innerhalb der Teilpopulationen und beschränkter Rekombination der Teilpopulationen untereinander benötigt man nur wenig Kommunikation zwischen den Prozessoren, so dass man einen fast linearen Speed-up erhält. Die Praxis zeigt, dass der Speed-up sogar oft superlinear ist, also schneller ansteigt als es der zusätzlichen Rechenleistung entspräche. Erklären lässt sich dies nur durch die Andersartigkeit des parallelen und des sequentiellen Algorithmus. Das Konzept der Teilpopulationen verhindert nämlich zuverlässiger als das der Gesamtpopulation, dass ein evolutionärer Algorithmus während der Suche in einem lokalen Optimum steckenbleibt.

Ein anderes Problem stellen die vielen Parameter des Verfahrens dar:

- die Grösse der Population
- die maximale Anzahl der Generationen
- die Auswahl der genetischen Operatoren
- die Wahrscheinlichkeiten, mit denen die genetischen Operatoren angewendet werden

So hat man bei der Selektion die Wahl zwischen vielen unterschiedlichen Selektionsmethoden, die meistens selbst noch einen oder mehrere Parameter haben. Ähnliches gilt für die Rekombination, wo man Mutationswahrscheinlichkeiten und Mutationsarten sowie Kreuzungswahrscheinlichkeiten und Kreuzungsmethoden festzulegen hat. Bevor man eine Lösung für sein Problem findet, muss man deshalb oft durch langwieriges Ausprobieren die für das Problem geeignete Parametereinstellung finden. Die inzwischen gesammelten Erfahrungen haben sich aber glücklicherweise zu einigen Faustregeln verdichtet, und in jüngster Zeit gibt es auch häufiger analytische Untersuchungen von evolutionären Algorithmen, die es ermöglichen, Einstellungen für die Parameter abzuleiten.

Evolutionäre Algorithmen als Oberbegriff

Der Begriff evolutionäre Algorithmen ist eigentlich ein Oberbegriff zu einer Vielzahl unterschiedlicher Algorithmen. Und tatsächlich stellt ja jede neue Wahl einer Fitnessfunktion und einer Codierungsform einen «neuen» Algorithmus dar. Einige wichtige Unterklassen evolutionärer Algorithmen sind

- Evolutionsstrategien
- genetische Algorithmen
- evolutionäres Programmieren und
- genetisches Programmieren

Die unterschiedlichen Bezeichnungen sind zum Teil historisch, zum Teil sachlich

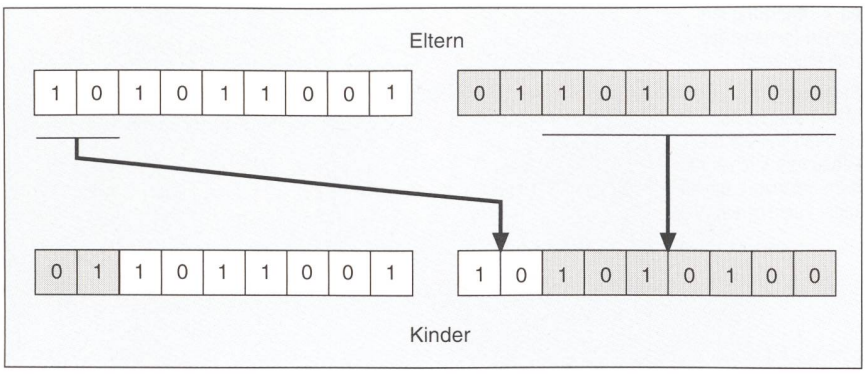

Bild 8 Kreuzung mit Bitstrings (genetische Algorithmen)
Die Strings der Eltern werden an der gleichen Stelle durchschnitten, und die Teilstücke werden ausgetauscht.

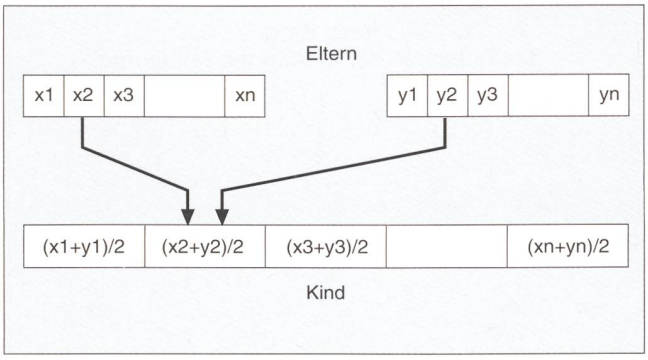

Bild 9 Kreuzung mit reellwertigen Vektoren (Evolutionsstrategien)
In diesem Beispiel der intermediären Vererbung entsteht nur ein Nachkomme. Der Wert ergibt sich aus der Mittelung der Werte der Eltern. Oft wird für jedes Vektorelement ein neues Elternteil bestimmt, so dass hier an der Kreuzung n+1 Individuen beteiligt sind.

bedingt. In den 70er Jahren wurden unabhängig voneinander in Deutschland und den USA Optimierungsverfahren auf der Basis der natürlichen Evolution entwickelt. Die genetischen Algorithmen wurden von John Holland in den USA zum erstenmal für sogenannte Klassifizierungssysteme (classifier systems) angewendet. Die Individuen werden dabei als Bitstrings repräsentiert, als Selektionsmethode wird proportionale Selektion eingesetzt, und der wesentliche Rekombinationsoperator ist die (zweigeschlechtliche) Kreuzung [2]. Demgegenüber sind die Evolutionsstrategien von Hans-Paul Schwefel und Ingo Rechenberg zur Parameteroptimierung entwickelt worden und arbeiten dementsprechend mit reellwertigen Vektoren als Individuen. Sie verwendeten hauptsächlich eine Art Truncation-Selektion und Mutationsoperatoren.

Inzwischen verwischen sich die Unterschiede, und die Ideen aus den unterschiedlichen Bereichen werden immer mehr zusammengeführt. Trotzdem halten sich die alten Bezeichnungen, und für erfolgreiche neue Varianten der evolutionären Algorithmen wird auch mal ein neuer Name eingeführt, wie zum Beispiel «genetisches Programmieren». Das Verfahren unterscheidet sich von den «genetischen Algorithmen» dadurch, dass die Individuen als Bäume dargestellt sind. Unser Fallbeispiel, der gut geparkte LKW, fällt also eigentlich unter die Kategorie «genetisches Programmieren». Das genetische Programmieren ist eben dieser Baumstruktur wegen so erfolgreich, sie ermöglicht es nämlich, ganze Computerprogramme als Optimierungsaufgabe zu betrachten, denn Computerprogramme lassen sich elegant als Bäume codieren. Damit lässt sich für eine ganze Problemklasse ein Lösungsprogramm finden, und eine Instanz des Problems kann dann durch das gefundene Lösungsprogramm gelöst werden. Da das Lösungsprogramm selbst keine «Genetik» mehr enthält, entfallen alle Nachteile der evolutionären Algorithmen (z.B. hohe Rechenzeit und hoher Speicherbedarf). Das ist ein wesentlicher Unterschied zu den anderen Varianten der evolutionären Algorithmen und zu den meisten anderen Optimierungsverfahren, wo für jede Probleminstanz ein neuer Optimierungslauf notwendig ist.

Fazit

Evolutionäre Algorithmen sind ein robustes, universell einsetzbares probabilistisches Optimierungsverfahren. Die benötigte hohe Rechenleistung ist vor allem dann gerechtfertigt, wenn wenig Wissen über die Lösung eines Problems vorhanden ist oder wenn man so viel wie möglich vom vorhandenen Wissen in den evolutionären Algorithmus einbringt, um die eingesetzte Rechenleistung sinnvoll zu nutzen. Obwohl das Verfahren sehr einfach zu beschreiben, zu implementieren und «intuitiv» zu verstehen ist, sollte die Analogie zur Natur nicht zu eng gesehen werden. Zumindest nicht solange man die evolutionären Algorithmen zur Optimierung einsetzt. Dann sollte man zur Analyse und Optimierung des Verfahrens auf mathematische und statistische Analysemethoden zurückgreifen.

Literatur

[1] *J. Bernasconi:* Simulated Annealing – eine Optimierungsmethode aus der statistischen Mechanik. Bulletin SEV/VSE 79(1988)21, S. 1295–1299.

[2] *David Goldberg:* Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company, 1989.

[3] *John Koza:* Genetic Programming. The MIT Press, 1992.

[4] *Zbigniew Michalewicz:* Genetic Algorithms + Data Structures = Evolution Programs. Artificial Intelligence. Springer, Berlin, 1992.

[5] *Lawrence Davis:* Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, 1991.

[6] *Thomas Bäck, Frank Hoffmeister, Hans-Paul Schwefel:* Applications of Evolutionary Algorithms. Technical Report Sys. 2/92. System Analysis Research Group, Department of Computer Science, University of Dortmund, D-44221 Dortmund, ISSN 0941-4568.

[7] *Larry L. Eshelman (Ed):* Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA6), Morgan Kaufmann Publishers, San Francisco, 1995.

[8] *Yuval Davidor, Hans-Paul Schwefel, Reinhard Männer (Eds):* Parallel Problem Solving from Nature – PPSN III. Lecture Notes in Computer Science 866, Springer-Verlag, Berlin, 1994.

Optimiser à l'exemple de la nature

Algorithmes évolutionnaires

L'ambition de comprendre les règles de la nature est toujours allée de pair avec le désir d'appliquer ses méthodes de manière utile aux créations humaines. Depuis quelques années on constate un grand intérêt à de tel procédés, surtout parce que les procédés classiques n'offrent encore aucune solution satisfaisante à nombre de problèmes. Outre les réseaux neuronaux qui ont fait les gros titres des illustrés, les algorithmes évolutionnaires appartiennent certainement aux domaines des plus intéressants de la recherche actuelle.