

Evolutionäre Algorithmen für Rucksackprobleme

Von Jens Gottlieb

Evolutionäre Algorithmen sind heuristische Suchverfahren, die auf dem Darwinischen Evolutionsprinzip "Survival of the fittest" basieren. Sie haben im letzten Jahrzehnt wegen ihres einfachen Grundprinzips, der hohen Parallelisierbarkeit und der breiten Anwendbarkeit großes Interesse sowohl in der Wissenschaft als auch in der Industrie hervorgerufen. Evolutionäre Algorithmen sind insbesondere auch bei Constraint-Optimierungsproblemen erfolgreich eingesetzt worden. Diese Probleme zeichnen sich dadurch aus, daß eine Lösung optimal bezüglich eines Bewertungskriteriums sein muß und zusätzlichen Restriktionen – den sogenannten Constraints – unterliegt. Entscheidend bei der Anwendung evolutionärer Algorithmen auf solche Probleme ist die Wahl geeigneter Techniken zur Behandlung der Constraints. Während die Auswahl dieser Techniken bislang meist durch praktische Erfahrung bzw. Testen erfolgt und somit nicht durch theoretische Untersuchungen geleitet ist, werden in dieser Arbeit Rucksackprobleme systematisch untersucht. Zentrale Eigenschaft ist, daß die globalen Optima am Rand der zulässigen Region des Suchraums liegen. Analytische und empirische Untersuchungen zeigen, daß der Erfolg verschiedener Constraint-Behandlungstechniken entscheidend von ihrer Fähigkeit abhängt, die evolutionäre Suche auf den Rand der zulässigen Region zu konzentrieren.

begin

Setze $t = 0$

Initialisiere $P(0)$

while not Abbruchkriterium **do**

begin

Selektiere Eltern $PE(t)$ aus $P(t)$

Generiere Nachkommen $PN(t)$ von $PE(t)$

Selektiere $P(t+1)$ aus $P(t)$ und $PN(t)$

Setze $t = t + 1$

end

end

Bild 1: Ablaufschema evolutionärer Algorithmen

Evolutionäre Algorithmen

Evolutionäre Algorithmen sind an dem biologischen Evolutionsprinzip orientierte heuristische Suchverfahren, die auf dem Wechselspiel von Variation und Selektion basieren [7,8]. Die Lösung zu einem gegebenen Problem wird durch eine populationsbasierte Suche im Raum aller möglichen Lösungskandidaten – dem Suchraum – angestrebt. Die einzelnen Individuen einer Population werden gemäß einer Fitneßfunktion bewertet, die die Qualität bezüglich der zugrundeliegenden Problemstruktur widerspiegelt. Durch die Selektion werden Individuen mit höherer Fitneß bevorzugt, die dadurch einen höheren Einfluß auf den weiteren Suchprozeß erlangen. Durch Variation von Individuen entstehen Nachkommen, die relevante Eigenschaften von ihren Eltern erben, wodurch eine Konzentration der Suche in der Umgebung von Individuen mit hoher Fitneß ermöglicht wird.

Der allgemeine Ablauf evolutionärer Algorithmen wird in **Bild 1** gezeigt. Die Population $P(t)$ wird anfangs ($t=0$) zufällig initialisiert und danach in jeder Generation wie folgt verändert: aus $P(t)$ wird eine Elternpopulation $PE(t+1)$ selektiert, die dann eine Nachkommenpopulation $PN(t+1)$ durch Variation erzeugt. Der Ersetzungsschritt definiert die neue Population $P(t+1)$ in Abhängigkeit von $P(t)$ und $PN(t+1)$. Bei dem Variationsschritt kommen Operatoren wie Mutation und Crossover zum Einsatz, die geringfügige Änderungen durchführen bzw. aus den Gemeinsamkeiten zweier Eltern Nachkommen erzeugen. Um einen Fortschritt bezüglich der Lösungsqualität zu erzielen, werden bei der Selektion Individuen mit höherer Fitneß bevorzugt. Der Evolutionsprozeß wird nach einer gewissen Zeit beendet und bringt durch die fortwährende Kombination von Variation und Selektion häufig Lösungskandidaten hoher Qualität hervor.

Grundsätzlich können evolutionäre Algorithmen direkt in dem gegebenen Suchraum suchen, falls geeignete Va-

riationsoperatoren verfügbar sind, oder eine indirekte Suche durchführen, indem ein alternativer Suchraum durchsucht wird, dessen Elemente mittels eines Decoders in den ursprünglichen Suchraum abgebildet werden. Der indirekte Ansatz ermöglicht es, Problemwissen in den Decoder zu integrieren und einfachere Variationsoperatoren für den neuen Suchraum zu benutzen, während der direkte Ansatz unter Umständen problemangepaßte Variationsoperatoren oder auch Penalty-Funktionen benötigt.

Rucksackprobleme

Ein Rucksackproblem besteht aus einer Menge von Gegenständen, die jeweils einen gewissen Profit einbringen und ein gewisses Gewicht haben, und einer Gewichtsbeschränkung. Gesucht ist nun eine Teilmenge der Gegenstände, die die Gewichtsbeschränkung einhält und einen maximalen Gesamtprofit liefert. Als Beispiel betrachten wir das Problem, bei dem aus einer Menge von $n=3$ Gegenständen eine Teilmenge durch die Entscheidungsvariablen (3) ausgewählt werden soll, die die Zielfunktion (1) maximiert und den Constraint (2) einhält. Der entsprechende Suchraum, der alle potentiellen Lösungskandidaten enthält, ist in **Bild 2** dargestellt, wobei unzulässige Lösungskandidaten grau gefärbt sind. Die Nachbarschaftsrelation verbindet Kandidaten, die bis auf eine Entscheidungsvariable identisch sind.

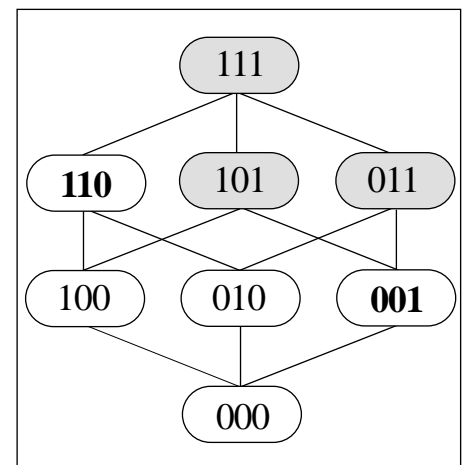


Bild 2: Der Suchraum des Rucksackproblems

In [2] werden einige generelle Eigenschaften des Suchraums aus der Monotonie des Constraints hergeleitet:

- Wenn x unzulässig und y oberer Nachbar von x ist, dann ist auch y unzulässig.
- Wenn y zulässig und x unterer Nachbar von y ist, dann ist x auch zulässig.

Wegen der strengen Monotonie der Zielfunktion folgt nun, daß das globale Optimum auf dem Rand der zulässigen Region liegt, der definiert ist als die Menge der zulässigen Lösungskandidaten, deren obere Nachbarn alle unzulässig sind; in **Bild 2** ist dieser Rand fett gedruckt und enthält die Kandidaten (1,1,0) und (0,0,1).

Covering- und Packing-Probleme werden in [2] als die Problemklasse definiert, die die Eigenschaft hat, daß die globalen Optima auf dem Rand der zulässigen Region liegen. Zu dieser Klasse gehören viele komplexe Probleme wie das Set-Covering-Problem und das mehrdimensionale Rucksackproblem, welches als Verallgemeinerung des eindimensionalen Falls mehreren Gewichtsbeschränkungen unterliegt und als repräsentatives Problem in den folgenden empirischen Untersuchungen verwendet wird. Viele praktische Fragestellungen wie z.B. Standortplanung, Ressourcenverteilung in Computernetzen oder Projektplanung lassen sich durch Covering- und Packing-Probleme modellieren [6,10]. Wegen ihrer NP-Vollständigkeit ist man in der Praxis auf effektive Heuristiken angewiesen, was insbesondere auch den Einsatz von evolutionären Algorithmen motiviert.

Direkte Suche im vollständigen Suchraum

Evolutionäre Algorithmen, die eine direkte Suche im vollständigen Suchraum durchführen, benutzen einfache Variationsoperatoren, die auch unzulässige Nachkommen erzeugen können. Die Bewertung von unzulässigen Individuen erfolgt durch eine Penalty-Funktion, die die Unzulässigkeit eines Individuums „bestraft“ und von der gegebenen Zielfunktion subtrahiert wird. Die in der Literatur für mehrdimensionale Rucksackprobleme vorgeschlagenen Penalty-Funktionen leiden meist unter dem Zulässigkeitsproblem [4,5], d.h. der evolutionäre Suchprozeß terminiert mit einer vollständig unzulässigen Population.

Bisherige Untersuchungen haben sich nur auf die empirische Auswertung der finalen Population gestützt und kaum Erkenntnisse zu den eigentlichen Ursachen des Zulässigkeitsproblems geliefert. In [2] dagegen wird eine analytische Untersuchung durchgeführt, bei der sich zeigt, daß durch ungünstig gewählte Penalty-Funktionen

1. neue lokale Optima im unzulässigen Teil des Suchraums entstehen und
2. ursprünglich globale Optima nicht mehr global optimal sind.

Bei solchen Penalty-Funktionen konvergiert ein evolutionärer Algorithmus in der unzulässigen Region. Durch eine gewisse Monotoniebedingung an Penalty-Funktionen kann jedoch garantiert werden, daß weder Fall 1 noch Fall 2 eintreten kann und somit die finale Population eines evolutionären Suchprozesses stets zulässige Lösungskandidaten beinhaltet. Empirische Untersuchungen haben gezeigt, daß die in [2] auf Basis der Monotoniebedingung entwickelte Penalty-Funktion die Population zuverlässig in den Rand der zulässigen Region führt, wo bekanntlich die globalen Optima liegen, und stets mit zulässigen Lösungskandidaten konvergiert, also das Zulässigkeitsproblem löst.

Die Dynamik des Suchprozesses kann durch das Hamming-Gewicht der einzelnen Populationen visualisiert werden. In Bild 3 ist die Dynamik für die [4] entnommene Funktion penalty6 sowie die neue Funktion penalty9 für verschiedene Initialisierun-

gen dargestellt. Für das betrachtete Problem ist bekannt, daß die Grenze der zulässigen Region etwa bei einem Hamming-Gewicht 120 liegt – oberhalb dieses Gewichts finden sich wie im vorigen Abschnitt beschrieben folglich nur unzulässige Lösungen. Unabhängig von dem Hamming-Gewicht der Startpopulation führt penalty6 die Population in die Region maximaler Zulässigkeit, die ein Hamming-Gewicht 500 hat, während penalty9 in der Lage ist, eine beliebige Startpopulation in die Grenzregion zu leiten. Dieses dynamische Verhalten kann anhand der Monotonie-Eigenschaften der Funktionen formal vorhergesagt werden: Während penalty9 die notwendige Monotoniebedingung er-

füllt, wird diese von penalty6 verletzt.

Direkte Suche im zulässigen Suchraum

Der Einsatz von Reparatur-Algorithmen erlaubt die Einschränkung der tatsächlich untersuchten Individuen auf den zulässigen Suchraum. Falls durch Variation ein unzulässiges Individuum erzeugt wird, kann dieses durch geeignete Veränderungen wieder zulässig gemacht werden. Eine weitere Verbesserung ergibt sich durch den zusätzlichen Einsatz von lokaler Optimierung [1,9].

In [2] werden neue Initialisierungsroutinen vorgeschlagen, die nur Lösungskandidaten erzeugen, ►

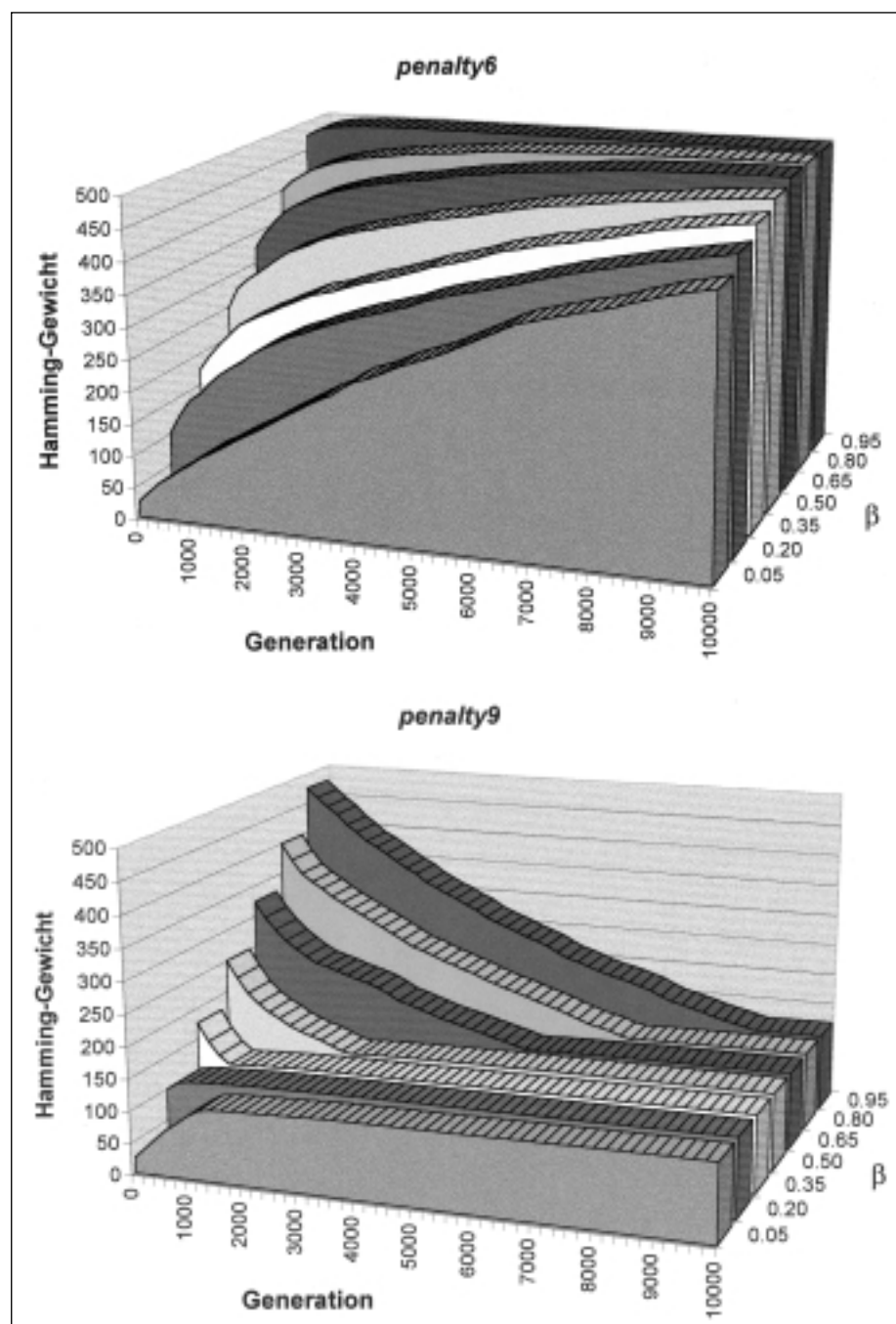


Bild 3: Die Dynamik der Hamming-Gewichte für penalty6 und penalty9 für verschiedene Initialisierungen

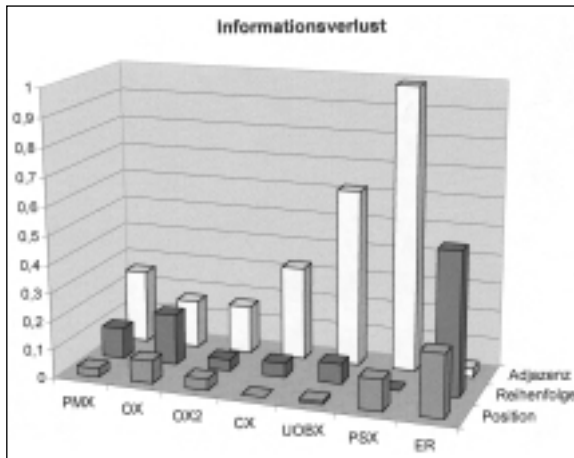


Bild 4: Informationsverlust bei verschiedenen Crossover-Operatoren bezüglich der Informationstypen Position, Reihenfolge und Adjazenz

den statistische Distanzmaße eingeführt, die sich auf die in Permutationen codierten Informationen bezüglich Reihenfolge, Adjazenz und absoluter Position der einzelnen Elemente beziehen und teilweise sogar analytisch bestimmt werden konnten.

Beispielhaft wird in **Bild 4** der Informationsverlust für verschiedene Crossover-Operatoren gezeigt, der angibt, wieviele gemeinsame Informationen der Eltern beim Crossover verlorengehen. Ein zu hoher Informationsverlust bezüglich relevanter Informationstypen bedeutet, daß der evolutionäre Algorithmus wichtige Informationen nicht ausnutzen kann, was zu schlechter Performance führt. Im Falle des Traveling-Salesman-Problems ist nur

die Adjazenz-Information wichtig, daher ist ER der beste Operator. Dagegen ist bei dem mehrdimensionalen Rucksackproblem Adjazenz irrelevant, während sowohl Positionen als auch Reihenfolge der Elemente eine Rolle spielen, so daß z.B. ER nicht adäquat und OX2 seinem Vorgänger OX überlegen ist.

Generell unterstützt die Studie von Permutationsoperatoren die Auswahl geeigneter Variationsoperatoren bei beliebigen Permutationsproblemen, sofern die relevanten Informationstypen bekannt sind.

Fazit

Während bislang meist empirische Resultate zu verschiedenen Constraint-Behandlungsmethoden in evolutionären Algorithmen vorlagen, sind in dieser Arbeit verschiedene Techniken sowohl analytisch als auch empirisch anhand einer speziellen Problemklasse untersucht worden. Die unterschiedlichen Verhaltensmuster sowie die erzielte Lösungsqualität konnten durch formale Analysen erklärt werden. Es ergab sich eine klare Differenzierung der oben behandelten grundlegenden evolutionären Vorgehensweisen, wobei sich der Einsatz von Reparatur-Algorithmen und lokaler Optimierung als besonders erfolgreich erwiesen hat.

Obwohl die empirischen Untersuchungen ausnahmslos anhand des mehrdimensionalen Rucksackproblems durchgeführt wurden, sind identische Resultate bei allen Covering- und Packing-Problemen zu erwarten, da als Verhaltensklärung der verschiedenen Techniken jeweils nur die dieser Problemklasse gemeinsame Eigenschaft benutzt wurde, die besagt, daß die globalen Optima auf dem Rand der zulässigen Region liegen.

Die Anwendung evolutionärer Algorithmen auf industrielle Probleme, die auf Covering- und Packing-Problemen basieren, wird durch diese Studie stark vereinfacht, da sowohl die erfolgversprechendsten Vorgehensweisen als auch typische

Fehler beim Design adäquater Constraint-Behandlungstechniken identifiziert wurden. Ein weiterer hoher praktischer Nutzen ergibt sich aus der allgemeinen Analyse von Permutationsoperatoren, die eine effiziente Auswahl von Variationsoperatoren ohne rechenaufwendige Simulationen gestattet. Die Charakterisierung gebräuchlicher Operatoren erleichtert die Implementierung evolutionärer Algorithmen für industrielle Fragestellungen, die sich auf Permutationsprobleme zurückführen lassen, wie z.B. Tourenplanungsprobleme oder Produktionsplanung.

LITERATUR

- [1] Chu, P.C.; J.E. Beasley: A Genetic Algorithm for the Multidimensional Knapsack Problem. *Journal of Heuristics*, Volume 4, No. 1, 63 - 86, 1998
- [2] Gottlieb, J.: *Evolutionary Algorithms for Constrained Optimization Problems*. Dissertation, Technische Universität Clausthal, Institut für Informatik, 1999. Shaker Verlag, Aachen, ISBN 3-8265-7783-3, 2000
- [3] Hinterding, R.: Mapping, Order-independent Genes and the Knapsack Problem. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, 13 - 17, 1994
- [4] Hoff, A.; A. Lokketangen; I. Mittet: *Genetic Algorithms for 0/1 Multidimensional Knapsack Problems*. In *Proceedings Norsk Informatikkonferanse*, 1996
- [5] Khuri, S.; T. Bäck; J. Heitkötter: *The Zero/One Multiple Knapsack Problem and Genetic Algorithms*. In *Proceedings of the ACM Symposium on Applied Computation*, 188 - 193, ACM Press, 1994
- [6] Martello, S.; P. Toth: *Knapsack Problems*. John Wiley & Sons, 1990
- [7] Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. Third Edition, Springer, 1996
- [8] Nissen, V.: *Einführung in Evolutionäre Algorithmen*. Vieweg, 1997
- [9] Raidl, G.R.: *An Improved Genetic Algorithm for the Multiconstrained 0 - 1 Knapsack Problem*. In *Proceedings of the 5th IEEE International Conference on Evolutionary Computation*, 207 - 211, 1998
- [10] Salkin, H.M.; K. Mathur: *Foundations of Integer Programming*. North-Holland, 1989

Anm. d. Red.:

Der Verfasser wurde für seine Dissertation mit dem Förderpreis des Vereins von Freunden 1999 ausgezeichnet.

Dr. Jens Gottlieb
SAPAG

Neurottr. 16

69190 Walldorf

Tel.: 06227/7-49356

Fax: 06227/78-32766

E-Mail: jens.gottlieb@sap.com

die auf dem Rand der zulässigen Region liegen. Neue theoretische Resultate zeigen, daß sowohl im Sinne von Qualität als auch Lokalität eine „perfekte“ Reparatur bzw. lokale Optimierung zu NP-vollständigen Teilproblemen führt, weshalb in [2] eine Reihe von Heuristiken untersucht werden.

Die empirischen Resultate zeigen, daß der konsequente Einsatz lokaler Optimierung die besten Ergebnisse erzielt und deterministische Heuristiken in Reparatur und lokaler Optimierung besser als ihre entsprechenden stochastischen Varianten sind. Insgesamt erweist sich diese Vorgehensweise als die erfolgreichste – bei mehr als 100 der 270 in [1] eingeführten Benchmarks wurde eine neue beste zulässige Lösung gefunden.

Indirekte Suche im zulässigen Suchraum

Im Gegensatz zu den bisher vorgestellten Ansätzen basiert die indirekte Suche auf einem neuen Suchraum, der mittels eines Decoders in die zulässige Region des ursprünglichen Suchraums abgebildet wird. Die in [3] vorgeschlagene Repräsentation decodiert eine Permutation der Zahlen 1,...,n durch einen einfachen Greedy-Algorithmus in eine zulässige Lösung für ein Rucksackproblem mit n Gegenständen, die sogar auf dem Rand der zulässigen Region liegt [2]. Eine Vielzahl an Mutations- und Crossover-Operatoren ist jeweils für spezielle Permutationsprobleme wie z.B. Scheduling oder das Traveling-Salesman-Problem vorgeschlagen worden [7]; jedoch ist es meist nicht offensichtlich, welcher Operator bei welchem Problem die beste Performance liefert. Bislang erfolgte die Operatorauswahl häufig durch Vergleiche anhand von rechenintensiven Studien, in denen evolutionäre Algorithmen mit verschiedenen Operatoren anhand der erzielten Lösungsqualität verglichen wurden.

In [2] wird eine formale Untersuchung verschiedener Variationsoperatoren vorgenommen, die eine Charakterisierung anhand relevanter Informationstypen erlaubt, ohne dabei aufwendige Läufe evolutionärer Algorithmen zu erfordern. Es wer-