Eine Übersicht über Crossover-Operationen für genetische Algorithmen
Seminar Organic Computing

Gerald Siegert
Matrikelnummer: 1450117

Universität Augsburg
Lehrstuhl für Organic Computing
student@organic-computing.org

Abstract. Crossover-Operationen (CO) sind ein wesentlicher Teil von Genetischen Algorithmen (GA) und sind maßgeblich für deren Effizienz. Daher soll ein kleiner Überblick über verschiedene COs und deren Klassifizierung gegeben werden. Zunächst werden eindimensionale Repräsentationen betrachtet. Dabei werden Darstellungen als Binärwerte, Ganzzahlen bzw. entsprechende Permutationen, Fließkommazahlen und Zeichenketten erläutert, bei welchen Problemen bzw. Anwendungsfällen welche Darstellung geeignet ist, und einige der dafür optimierten COs aufgezeigt. Ebenso betrachtet werden mehrdimensionale Repräsentationen wie Bäume und Arrays. Es wird zudem eine kleine Übersicht über weitere mehrdimensionale Repräsentationen gegeben. Ebenso wird auch darauf eingegangen, wann es geeignet ist, anwendungsspezifische Codierungen zu nutzen und anzuwenden. Ebenso werden zudem einige universell nutzbare COs aufgezeigt, die nicht an eine spezielle Repräsentation der Daten gebunden sind.

1 Einführung in genetische Algorithmen

Genetische Algorithmen (GA) sind genauso wie andere Evolutionäre Algorithmen im Allgemeinen aus der Biologie übernommen worden. Wie der Name schon aussagt, basieren sie auf dem Prinzip der Evolution, bei der basierend auf einer Ausgangspopulation möglicher Lösungen neue Kinder erzeugt werden, welche dann die Vorfahren in der Population verdrängen. Welche Vorfahren, oder gar die erzeugten Kinder, dabei konkret verdrängt werden, entscheidet sich basierend auf einer Fitness-Funktion, bei der die gefundenen Lösungen der Population evaluiert werden und anschließend nur die besten in der Population verweilen dürfen.

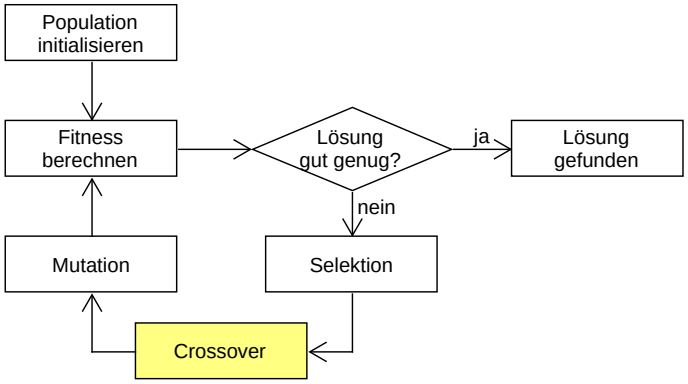

Fig. 1: Grundlegender Ablauf eines genetischen Algorithmus

Die wichtigen Parameter eines GA selbst sind zum einen die Selektion der Gene, deren Crossover-Operationen (CO) zur Erzeugung neuer Kinder, sowie die Durchführung anschließender Mutationen. Maßgeblich für die Qualität und Effizienz eines GA ist dabei die in Fig. 1 markierte CO.

In dieser Seminararbeit soll daher nun ein kleiner Überblick über einige verschiedene COs gegeben werden. Nach einer Übersicht der Klassifikationen im Abschnitt 2 werden im Abschnitt 3 zuerst geeignete Anwendungen und dazugehörige COs für eindimensionale, im darauf folgenden Abschnitt 4 eine Übersicht für mehrdimensionale Repräsentationen aufgezeigt. Anschließend wird im Abschnitt 5 ein kurzer Überblick über anwendungsspezifische Codierung, sowie im Abschnitt 6 über universell einsetzbare COs gegeben.

2 Klassifizierungen von Crossover-Operationen

COs gibt es für viele verschiedene Arten von Anwendungen und Daten. Jedoch sind nicht alle möglichen Operationen für jede Anwendung und Daten-

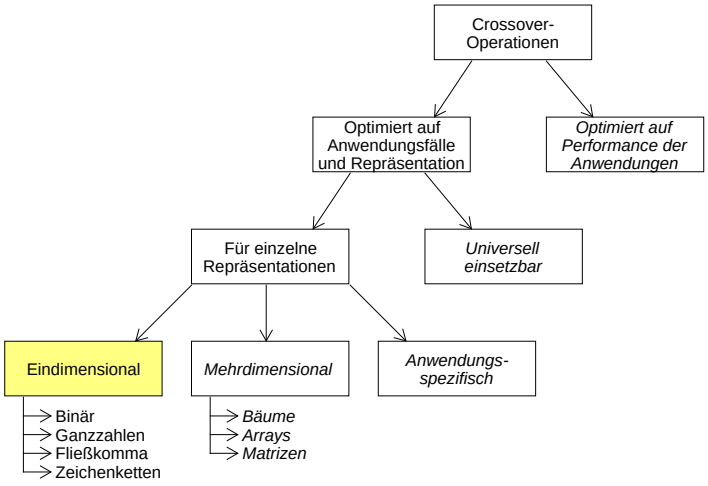

Fig. 2: Übersicht der Klassifizierung (nach [18])

art geeignet. Pavai und Geetha haben daher in [18] bereits einen sehr umfassenden Überblick über die verschiedenen Arten von COs gegeben, weshalb sich diese Seminararbeit an deren Arbeit orientiert. COs können, wie in Fig. 2 dargestellt, in verschiedenen Kategorien klassifiziert werden. Diese Arbeit befasst sich hauptsächlich mit eindimensionalen Darstellungsformen der Daten und deren entsprechenden COs. Für die kursiv markierten Klassifizierungen wird daher nur ein kurzer Überblick bzw. im Falle der performance-optimierten COs darauf verzichtet.

Die Basis aller verschiedenen COs bilden die elementaren Operationen, welche die Gene der beiden Eltern an einer oder mehreren Stellen teilen und daraus die neuen Kinder erzeugen. Entsprechend werden sie auch *N-Point-Crossover* bzw. im Speziellen auch *One-Point-Crossover* oder *Two-Point-Crossover* genannt. Darauf basierend gibt es noch weitere Basis-Operationen, wie *Segmented Crossover*, bei der die Gene in eine bestimmte Anzahl Segmente anstatt an einer bestimmten Anzahl an Stellen geteilt werden, [22] oder wie *Uniform Crossover*, bei der für jede Stelle des Kindes zufällig ausgewählt wird, welcher Elternteil sein Gen vererbt.

3 Eindimensionale Repräsentation

Unter eindimensionaler Repräsentation wird vor allem die Darstellung der Daten in den elementaren Datentypen verstanden. Die Daten liegen dabei nur linear in einer bestimmten Reihenfolge vor, wodurch die Handhabung mit den Daten auch entsprechend einfach ist und in vielen Anwendungsfällen ohne große Probleme durchgeführt werden kann und entsprechend oft genutzt wird.

Zuerst wird im Abschnitt 3.1 ein Überblick über Anwendungsfälle und mögliche COs für die Binär-Codierung gegeben. Anschließend wird in Abschnitt 3.2 auf ganzzahlige Codierung allgemein bzw. auf Permutationen im Speziellen eingegangen, in Abschnitt 3.3 folgen Fließkomma-Darstellungen sowie in 3.4 Zeichenketten.

3.1 Binäre Codierung

Eine Codierung als Binärwerte bedeutet, dass die Werte, die vom GA bearbeitet werden, als eine Kette von 0 und 1 dargestellt sind. Der große Vorteil einer binären Codierung liegt vor allem darin, dass die Handhabung entsprechender Daten sehr einfach und platzsparend ist, weshalb diese Art der Darstellung prinzipiell von jeder Anwendung genutzt werden kann. Ebenfalls ein großer Vorteil liegt darin, dass entsprechende GAs aufgrund des geringen Alphabets sehr schnell und effizient sind. [8]

Vor allem folgende Arten von Anwendungen sind besonders dafür geeignet, mit binärer Codierung zu Arbeiten: [18]

Classification Problem Lösung soll in verschiedene Kategorien klassifiziert werden. [9]

Multimodal Spin Lattice Problem Suchen eines minimalen Energiezustandes für 450 Spins mit je vier Zuständen auf einem 2D-Gitter. [16]

Passende Crossover-Operationen dafür sind:

Self-Crossover Tauschen von einzelnen Bits innerhalb des Chromosoms. [17]

Supplementary Crossover Nutzt das *Center of Gravity*-Paradigma um Kinder zu erzeugen. [3]

Generalized crossover Interpretiert Chromosom als Ganzzahl und dividiert diese durch eine andere, zufällig ausgewählte Ganzzahl. [7]

3.2 Codierung als Ganzzahlen und Permutationen

Da ganzzahlige Werte ebenfalls sehr einfach als Binärwerte dargestellt werden können, können COs für Binärwerte prinzipiell auch für ganzzahlige Werte eingesetzt werden. Natürlich gibt es auch entsprechende Anwendungsfälle und COs, die speziell für ganzzahlige Darstellungen geeignet und optimiert sind, auf die hier aber verzichtet wird und stattdessen auf Ketten von ganzzahligen Werten, also Permutationen, eingegangen wird.

Der Unterschied zwischen einfacher ganzzahligen Darstellungen und Permutationen liegt darin, dass bei Permutationen die komplette Zahlenfolge als mehrere aneinandergereihte Zahlen betrachtet werden muss. Entsprechend können nicht einfache COs für Binärwerte und einfache ganzzahlige Werte eingesetzt werden, sondern COs, welche die Zahlenketten entsprechend berücksichtigen. Dies ist u. A. bei folgenden Anwendungen und Problemen der Fall:

Traveling Salesman Problem (TSP) Mehrere Orte, die mit einer möglichst kurzen Strecke miteinander verbunden werden müssen. [1]

Graph Coloring Problem Knoten eines Graphen mit möglichst wenig und verschiedenen Farben einfärben. [14]

Quadratic Assignment Problem Summe der Distanzen zwischen Punkten minimieren, ähnlich wie TSP, nur mit quadratischer Kostenfunktion. [13]

Passende COs für Permutationen lassen sich in verschiedene Kategorien einteilen, welche sich basierend auf dem grundlegenden Vorgehen der COs klassifizieren lassen. Im Folgenden werden nun einige positionsbasierte, kantenbasierte und folgenbasierte COs vorgestellt. Daneben gibt es aber auch noch weitere, wie z. B. teilmengenbasierte, die auf Basis von Genom-Teilmengen arbeiten, *Cut and Splice*-basierte, die einen Teil der Elterngenome tauschen, oder distanzbasierte COs, bei denen die Anzahl der unterschiedlichen Gene zwischen allen Elternteile gleich sein muss.

Positionsbasierte COs beeinflussen die Gene ihrer Kinder basierend auf den Positionen der einzelnen Gene. Dies sind u. A. folgende COs:

Partially Mapped Crossover (PMX) Wählt zufällig zwei Punkte aus und tauscht alle Werte dazwischen mit dem anderen Elternteil. Bei der Erweiterung Uniform PMX werden einzelne Gene anstatt eines ganzen Segments getauscht. [1] [12] [13]

Position Based Crossover (POS) Wählt einige zufällige Positionen im ersten Elternteil aus und verschiebt die ausgewählten Werte zu den korrespondierenden Positionen im anderen Elternteil. [12]

Kantenbasierte COs erzeugen basierend auf Kanten der Eltern(-knoten) neue Kinder. Darunter fallen u. A. folgende COs:

Edge Crossover Wählt einen Knoten mit der geringsten Kantenzahl aus und fügt ihn zum neuen Kind hinzu und löscht ihn aus den anderen Kantenlisten. [12]

Edge Exchange Crossover (EXX) Vererbt zunächst alle Elternkanten um einen Kreis zu bilden, entfernt dann aber ungültige Kanten. Falls kein Kreis gebildet werden konnte, werden alle Kanten vererbt. [26]

Folgenbasierte COs beeinflussen vor allem die Reihenfolge der Gene. Sie lassen sich in weitere Unterkategorien einteilen. Darunter fallen z. B. folgende COs:

Merging Crossover (MOX) Beide Eltern werden zuerst zufällig zusammengefügt und anschließend in die beiden Kinder geteilt. [14]

Non-Wrapping Ordered Crossover Erstellt Lücken und füllt diese wieder auf, ohne dabei die absolute Reihenfolge der Gene zu verliern. [1]

Daneben gibt es auch noch weitere COs, welche sich neben den hier vorgestellten Unterkategorien (verschmelzende und absolute Reihenfolge) z. B. in sortierende oder angrenzende, folgenbasierte COs einsortieren lassen.

3.3 Codierung als Fließkommazahl

Im Gegensatz zur Codierung als Ganzzahlen bzw. Binärwerte lassen sich Daten, welche als Fließkommazahlen codiert sind, viel genauer Darstellen, haben dadurch aber auch einen deutlich größeren Suchraum für mögliche Lösungen. Dennoch liegt gerade darin der Vorteil, da die Art der Datencodierung möglichst mit der Codierung des Suchraumes übereinstimmen sollte. Dadurch entfällt die notwendige Konvertierung der Daten, wodurch letztlich die Geschwindigkeit und damit die Effizienz des GAs deutlich erhöht wird. Ebenso ist von Vorteil, dass große Suchräume mit kontinuierlichen Daten auf einfache Art und Weise abgesucht werden können anstatt lediglich mit diskreten Daten wie bei der Binärcodierung. [8]

Entsprechend lassen sich vor allem Anwendungen bzw. Probleme, welche einen kontinuierlichen Suchraum besitzen, mit Fließkomma-Codierungen effizient ausführen. Darunter fallen z. B. folgende Anwendungen und Probleme:

Animal Diet Formulation Problem Erstellung eines Ernährungsplans mit möglichst vielen Nährstoffen zu minimalen Kosten. [19]

Electromagnetic Optimization Handhabung und Optimierung von elektromagnetischen Werten in einem kontinuierlichem Suchraum. [15]

Revenue Management in Airlines Zuweisen von begrenzten Ressourcen einer Airline (Flugzeuge, Personal, ...), um den Gewinn zu maximieren. [21]

Als Fließkommazahlen codierte Daten kann man auf mehrere Arten verarbeiten, mit deren Basis die COs funktionieren. Dies sind z. B. folgende Arten bzw. COs:

Taguchi Crossover (TC) Nutzt statistische Daten innerhalb einer Matrix um das beste Ergebnis zu finden. [5]

Average Crossover Modifizierter *One-Point-Crossover*, bei dem die Durchschnittswerte zwischen beiden Eltern anstatt den Elternwerten der einzelnen Gene genutzt werden. [19]

Fuzzy Arithmetic Weighted Mean (FAWM) Berechnet mithilfe der Fuzzy-Arithmetik die Werte der Kinder. [21]

Daneben gibt es aber auch noch einige weitere Arten wie z. B. den gewichteten Durchschnitt oder dem *Center of Mass Crossover (CMX)*, welcher den gleichnamigen Ansatz nutzt um mithilfe des Massenzentrums der Eltern neue Kinder zu erzeugen. [27] Daneben kann man aber auch verschiedene COs nutzen, um die Kinder zu erzeugen.

3.4 Codierung als Zeichenkette

Bei einer Codierung der Daten als eine Zeichenkette werden die einzelnen Gene als Buchstaben codiert. Dies eignet sich daher entsprechend vor allem für textbasierte Anwendungen. Aber auch das **DNA Sequencing Problem** lässt sich

damit einfach bearbeiten. Dabei werden die einzelnen DNA-Proteine als Buchstaben bezeichnet, wodurch sich eine Zeichenkette ergibt, die mithilfe von entsprechenden COs einfach bearbeitet werden kann. [18]

Bei einer Zeichenketten-Darstellung der Daten muss man zwischen einer festen Länge der Zeichenkette und einer variablen Länge unterscheiden. Abhängig ist dies vor allem von der Länge der beiden Eltern, also ob die Zeichenketten der Eltern die gleiche oder eine unterschiedliche Länge besitzen.

Zeichenketten mit festen Längen gelten als einfacher zu Handhaben, weshalb diese Art der Repräsentation deutlich häufiger genutzt wird. Grund hierfür liegt darin, dass nicht entschieden werden muss, welche Gene hinzugefügt oder entfernt werden müssen. Jackson hat in seiner Masterarbeit [10] folgende beiden COs vorgestellt, welche man dafür nutzen kann:

C1 Crossover Ermittelt Sekundärstruktur-Elemente und legt die Schnittpunkte auf unstrukturierte Regionen und trennt somit unterschiedliche Strukturtypen.

C2 Crossover Legt die Schnittpunkte basierend auf der Frequenz der angrenzenden Genompaare der Sekundärstruktur fest.

Zeichenketten mit variablen Längen werden dagegen häufig vor allem im Bereich der biologischen Anwendungen wie die Bearbeitung von DNA- bzw. RNA-Sequenzen genutzt. Dies liegt darin Begründet, dass sich diese Art der Darstellung auch in der Natur wiederfindet, wo Chromosomen ebenfalls unterschiedliche Längen haben. Bei Zeichenketten mit variablen Längen liegt eines der Hauptprobleme in der Entscheidung, welche Gene hinzugefügt und welche Gene entfernt werden sollen, wobei die Gesamtanzahl der Gene der Kinder der Anzahl aller Elterngene entsprechen soll. Dazu gibt es z. B. folgende COs, welche hierfür genutzt werden können:

Internal Crossover Basiert auf der Annahme, dass Gene der kleineren Sequenz ohne Überhang mit jedem Block der längeren Sequenz angeordnet werden können. [4]

Equal Crossover Erzeugt Kinder, welche die gleiche Längen wie die Eltern besitzen. [24]

Inside Crossover Erzeugt Kinder, die länger als der kurze und kürzer als der lange Elternteil sind. [24]

4 Mehrdimensionale Repräsentation

Neben einer eindimensionalen Repräsentation der Daten gibt es natürlich auch mehrdimensionale Repräsentationen. Damit sind nicht einfache lineare, sondern komplexere Datenstrukturen gemeint, die entsprechend komplizierter zu Handhaben sind. Da es aber auch Anwendungen gibt, die mehrdimensionale Daten verarbeiten, gibt es auch entsprechende COs, um solche Daten einfach verarbeiten zu können.

Arrays dürften wohl eine der einfachsten Arten der Mehrdimensionalen Repräsentationen darstellen. Sie bilden nicht mehr als eine Verkettung von mehreren linearen Datensätzen. Diese Art der Codierung kann für verschiedene Probleme genutzt werden wie z. B. das TSP oder die Suche nach Pareto-Optimaler Lösungen, bei denen nicht alle Zieleigenschaften optimal sein können.

Die dazu passenden COs heißen u. A. *Orthogonal Array Based Crossovers*, von denen es zwei Versionen gibt. Die erste Version heißt *Orthogonal Crossover* (CO) und erzeugt die Kinder, indem die Elterngene basierend auf den Kombinationen eines orthogonalen Array ausgewählt werden. Die zweite Version namens *Main Effect Crossover* (MC) wählt die Kindergene im orthogonalen Array dagegen basierend auf deren Effekte aus, wodurch Kinder mit den besten Haupteffekten der Eltern generiert werden. [6]

Weitere mehrdimensionale Repräsentationen sind beispielsweise *Bäume* oder *Matrizen*, bei denen die Daten entsprechend strukturiert und aufgebaut sind. Beide Darstellungsarten lassen sich u. A. auch durch ähnliche COs bearbeiten, wie z. B. der *Reijmers Crossover Operator*, welcher aus den Distanzen zwischen den einzelnen Baumknoten eine Distanzen-Matrix erstellt, und diese mit einer vom GA generierten *Correction-Matrix* addiert. [20]

5 Anwendungsspezifische Codierung der Daten

Natürlich muss man nicht immer eine der üblichen ein- oder mehrdimensionalen Repräsentationsformen nutzen, wenn man einen GA entwickeln möchte. Es gibt auch Anwendungsfälle wie z. B. in der Robotik, [11] bei denen es geeigneter ist, eine speziell für die Anwendung zugeschnittene und optimierte Daten-Codierung zu nutzen.

Die einfachste Variante dafür ist eine Art hybride Codierung zu nutzen, bei der zwei lineare Darstellungsformen miteinander kombiniert werden. Aguilar-Ruiz et al. haben dies in ihrer Arbeit [2] gemacht und diskrete und kontinuierliche Darstellungsformen miteinander kombiniert, unterscheiden ihre *Natural Encoding* jedoch von anderen hybriden Codierungen. Dennoch funktioniert ihr dabei entwickelter GA für klassische hybride und die *Natural Encoding*.

Daneben gibt es aber noch zahlreiche weitere Möglichkeiten, die Daten anwendungsspezifisch zu codieren und zu Handhaben. Einige davon wären z. B. *Restricted Growth Function*-basierte Techniken und COs, [25] oder die Möglichkeit, Fuzzy-Logiken zu nutzen. [23]

6 Universale Crossover-Operationen

Neben den in dieser Seminararbeit vorgestellten spezifischen Darstellungsformen und entsprechenden COs gibt es aber auch noch einige COs, welche nicht auf eine bestimmte Repräsentationsart beschränkt sind, sondern mehrere verschiedene auf gleiche Art und Weise bearbeiten und daraus neue Kinder erzeugen können.

Dies trifft zwar auch auf alle elementaren COs wie die *N-Point-Crossover* zu, aber auch auf zahlreiche weitere COs, wie z. B. der in [12] erwähnte *Sorted Match Crossover*. Diese CO basiert auf den Kosten bzw. Fitness der Eltern und erzeugt neue Kinder, indem er einen Teil eines teuren Elternteils mit einem Teil eines billigen Elternteils kombiniert.

Ähnlich wie der *Uniform Crossover Operator* gibt es auch noch weitere COs, bei denen die Gene der Eltern zufallsbasiert vererbt werden. Ziemlich ähnlich funktionieren auch der *Box Crossover* bzw. *Extended Box Crossover*. [28]

Zudem besteht auch noch die Möglichkeit, zwei vorhandene COs miteinander zu kombinieren, wobei hier eine aufeinanderfolgende oder eine voneinander unabhängige (Hybride) Ausführung möglich sind. Alternativ besteht auch die Möglichkeit, Heuristik-Basierte COs zu nutzen. Dadurch können auch ursprünglich für eine bestimmte Darstellungsart entwickelte COs universell eingesetzt werden, was die Performance und Qualität der gefundenen Lösungen nicht negativ beeinflussen muss.

7 Zusammenfassung

Neben den hier vorgestellten COs gibt es für alle Darstellungsformen und deren Kategorisierungen noch zahlreiche weitere COs, von denen aufgrund der Länge dieser Seminararbeit viele nicht mal im Ansatz erwähnt wurden bzw. konnten. Wie Anfangs erwähnt haben Pavai und Geetha in [18] eine Umfassende Übersicht erstellt, die sich ebenfalls lohnt, wenn man auf der Suche nach einer passenden CO für seine Anwendung ist.

Natürlich ist nicht jede der vorgestellten Darstellungsformen für alle Anwendungen geeignet, so macht es z. B. wenig Sinn, eine TSP-Anwendung mithilfe von Binärcodierung zu entwickeln, obwohl sich diese Codierungsart grundsätzlich für jede Anwendung eignet. Dies liegt zum einen natürlich mit dem dadurch erhöhtem Aufwand zur Konvertierung der Daten zusammen, zum anderen aber auch mit dem Grundsatz, dass sich der Lösungsraum und der Suchraum möglichst gering unterscheiden sollten. [8] Als Konsequenz daraus sind solche COs, welche direkt den Lösungsraum durchsuchen, am effizientesten.

Wichtig ist daher, dass zunächst ermittelt wird, welche Art der Repräsentation für die Daten am geeignetsten ist. Einige Beispiele für Anwendungen wurden für die einzelnen Darstellungsarten bereits genannt, jedoch gibt es auch noch zahlreiche weitere, welche für eine Darstellungsart passend sind. Darauf basierend kann danach entschieden werden, welche CO konkret genutzt wird, wobei auch hier jede Vor- und Nachteile besitzt, wodurch auch nicht jede CO immer geeignet ist. Es ist dabei auch immer eine Frage der Performance, Komplexität und Effizienz, welche Konfiguration für die Anwendung geeignet ist.

References

1. Abdoun, O., Abouchabaka, J.: A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem. CoRR abs/1203.3097 (2012), http://arxiv.org/abs/1203.3097
2. Aguilar-Ruiz, J.S., Giraldez, R., Riquelme, J.C.: Natural encoding for evolutionary supervised learning. IEEE Transactions on Evolutionary Computation 11(4), 466–479 (Aug 2007)
3. Angelov, P.: Supplementary crossover operator for genetic algorithms based on the center-of-gravity paradigm. Control and Cybernetics Vol. 30, no 2, 159–176 (2001)
4. Baake, M.: Repeat distributions from unequal crossovers. ArXiv e-prints (Mar 2008)
5. Chan, K.Y., Aydin, M.E., Fogarty, T.C.: A taguchi method-based crossover operator for the parametrical problems. In: Evolutionary Computation, 2003. CEC '03. The 2003 Congress on. vol. 2, pp. 971–977 Vol.2 (Dec 2003)
6. Chan, K., Kwong, C., Jiang, H., Aydin, M., Fogarty, T.: A new orthogonal array based crossover, with analysis of gene interactions, for evolutionary algorithms and its application to car door design. Expert Systems with Applications 37(5), 3853 – 3862 (2010), http://www.sciencedirect.com/science/article/pii/S0957417409009750
7. Coli, M., Gennuso, G., Palazzari, P.: A new crossover operator for genetic algorithms. In: Proceedings of IEEE International Conference on Evolutionary Computation. pp. 201–206 (May 1996)
8. Herrera, F., Lozano, M., Verdegay, J.: Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. Artificial Intelligence Review 12(4), 265–319 (1998), http://dx.doi.org/10.1023/A:1006504901164
9. Ho, S.Y., Liu, C.C., Liu, S.: Design of an optimal nearest neighbor classifier using an intelligent genetic algorithm. Pattern Recognition Letters 23(13), 1495 – 1503 (2002), http://www.sciencedirect.com/science/article/pii/S0167865502001095
10. Jackson, D.: Modified crossover operators for protein folding simulation with genetic algorithms. Master's thesis, Carleton University Ottawa (Sep 2004)
11. Koza, J.R.: Genetic programming: on the programming of computers by means of natural selection, vol. 1. MIT press (1992)
12. Kumar, R., Gopal, G., Kumar, R.: Novel crossover operator for genetic algorithm for permutation problems 3 (May 2013)
13. Misevičius, A., Kilda, B.: Comparison of crossover operators for the quadratic assignment problem. In: Information technology and control. vol. 34 (2005)
14. Mumford, C.L.: New Order-Based Crossovers for the Graph Coloring Problem, pp. 880–889. Springer Berlin Heidelberg, Berlin, Heidelberg (2006), http://dx.doi.org/10.1007/11844297_89
15. Otevrel, V., Raida, Z.: Mean-adaptive real-coding genetic algorithm and its applications to electromagnetic optimization (part one). vol. 16, p. 19. CZECH TECHNICAL UNIVERSITY (2007)
16. Pál, K.F.: Selection schemes with spatial isolation for genetic optimization, pp. 169–179. Springer Berlin Heidelberg, Berlin, Heidelberg (1994), http://dx.doi.org/10.1007/3-540-58484-6_261
17. Pal, N.R., Kundu, M.K., Nandi, S.: Application of a new genetic operator in feature selection problems. In: TENCON '98. 1998 IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control. vol. 1, pp. 37–40 vol.1 (1998)

18. Pavai, G., Geetha, T.V.: A survey on crossover operators. ACM Comput. Surv. 49(4), 72:1–72:43 (Dec 2016), http://doi.acm.org/10.1145/3009966
19. Rahman, R.A., Ramli, R.: Average concept of crossover operator in real coded genetic algorithm. vol. 63, pp. 73–77. IACSIT Press, Singapore (2013), https://search.proquest.com/docview/1522282250?accountid=8429, copyright - Copyright IACSIT Press 2013; Dokumentbestandteil - Tables; ; Equations; Zuletzt aktualisiert - 2014-05-08
20. Reijmers, T., Wehrens, R., Daeyaert, F., Lewi, P., Buydens, L.: Using genetic algorithms for the construction of phylogenetic trees: application to g-protein coupled receptor sequences. Biosystems 49(1), 31 – 43 (1999), http://www.sciencedirect.com/science/article/pii/S0303264798000331
21. Sadeghi, M., Sadeghi, R., Mousavi, S., Khanmohammadi, S.: Improved genetic algorithms by means of fuzzy crossover operators for revenue management in airlines. World Applied Sciences Journal 2(6), 838–846 (2013)
22. Sharapov, R.R.: Genetic Algorithms: Basic Ideas, Variants and Analysis. InTech (Jun 2007), https://www.intechopen.com/books/vision_systems_segmentation_and_pattern_recognition/genetic_algorithms__basic_ideas__variants_and_analysis
23. Sharma, S.K., Irwin, G.W.: Fuzzy coding of genetic algorithms. IEEE Transactions on Evolutionary Computation 7(4), 344–355 (Aug 2003)
24. Stringer, H., Wu, A.S.: Bloat is unnatural: An analysis of changes in variable chromosome length absent selection pressure. Univ. Central Florida, Tech. Rep. CS-TR-04-01 (2004)
25. Swift, S., Tucker, A., Crampton, J., Garway-Heath, D.: An improved restricted growth function genetic algorithm for the consensus clustering of retinal nerve fibre data. In: Proceedings of the 9th annual conference on Genetic and evolutionary computation. pp. 2174–2181. ACM (2007)
26. Takahashi, R.: A methodology of extended changing crossover operators to solve the traveling salesman problem. In: 2008 Fourth International Conference on Natural Computation. vol. 1, pp. 263–269 (Oct 2008)
27. Tsutsui, S., Ghosh, A.: A study on the effect of multi-parent recombination in real coded genetic algorithms. In: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360). pp. 828–833 (May 1998)
28. Yoon, Y., Kim, Y.H.: The Roles of Crossover and Mutation in Real-Coded Genetic Algorithms, Bio-Inspired Computational Algorithms and Their Applications, pp. 65–82. InTech (Mar 2012), https://www.intechopen.com/books/bio-inspired-computational-algorithms-and-their-applications/the-roles-of-crossover-and-mutation-in-real-coded-genetic-algorithms