Fundamente der Computational Intelligence

Dozent: Günter Rudolph
Vertretung: Nicola Beume

Wintersemester 2006/07
Universität Dortmund
Fachbereich Informatik
Lehrstuhl für Algorithm Engineering (LS11)
Fachgebiet *Computational Intelligence*

20.11.2006

Themengebiet II:
Optimierung mittels evolutionärer Algorithmen (EA)

Themen der heutigen Vorlesung:

- Was ist ein Optimierproblem?
- Einführung in evolutionäre Algorithmen

Beschreibung: Optimierproblem

Problem mit verschiedenen Lösungen
suche einen Pullover
Güte von Lösungen muss bewertbar sein
suche billigsten Pullover $\rightarrow$ *Bewertung in Euro*
suche besten Pullover $\rightarrow$ *Bewertung?*
Optimierung der Güte als mathematische Funktion:
Zielfunktion $g(\mathbf{x})$ zu optimieren

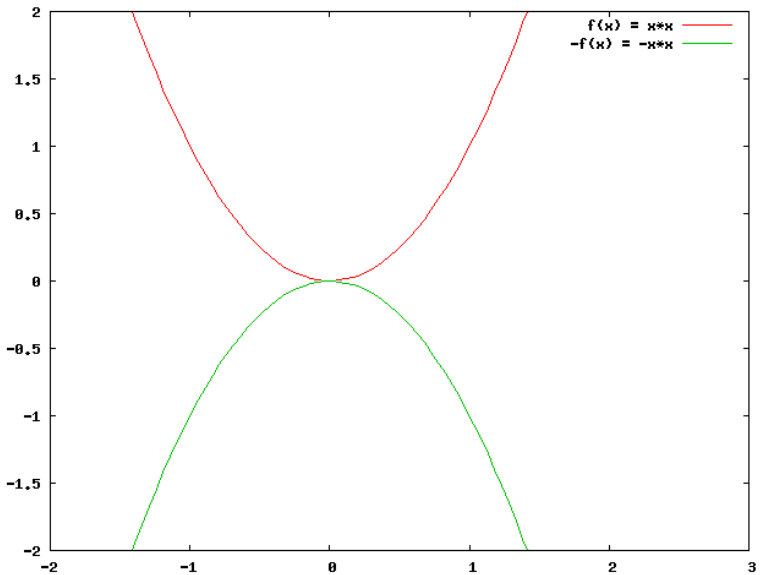
Allgemeinere Bezeichnung: Suchproblem, algorithmisches Problem
Gesucht ist eine Antwort auf die formalisierte Problemstellung

Andere Suchprobleme
Entscheidungsprobleme: Antworten ja/nein
Approximationsprobleme: Berechne Lösung mit bestimmter angenäherter Qualität

Definition: Optimierproblem

Zielfunktion $g(\mathbf{x})$ zu optimieren
Lösungsvektor $\mathbf{x} = (x_1, \ldots, x_n)$
Definitionsbereich von x_i z.B. $[u_i, o_i] \subset \mathbb{R}$
Wertebereich muss zumindest partiell geordnet sein, um Güte von Lösungen vergleichen zu können.

Minimierung analog zu Maximierung:
$g(\mathbf{x}) = \min! \iff -g(\mathbf{x}) = max!$



Eventuell Nebenbedingungen gestellt an $\mathbf{x}$
nicht alle Lösungen aus Definitionsbereich zulässig

Optimierprobleme und Lösungsverfahren

Lineare Optimierprobleme:
- lineare Zielfunktion, lineare Nebenbedingungen
- lösbar mit z.B. Simplex-Algorithmus

Nichtlineare Optimierprobleme:
- Zielfunktion oder Nebenbedingung nichtlinear
- lösbar mit konventionellen Verfahren, falls
- differenzierbar und
  - konvex (konvexe Zielfunktion, konvexer Definitionsbereich)
  - oder ohne Nebenbedingungen
  - (weitere Spezialfälle)

Sonstige Probleme

nicht lösbar mit konventionellen Verfahren

$\Rightarrow$ Stochastische Optimierverfahren, Computational Intelligence

Optimierung im Alltag

Alltagsproblem:
Schnellster Weg von zu Hause zur Uni?

Irgendeinen Weg ausprobieren.
Zeit messen.

Weg leicht verändern
Ausprobieren und Zeit messen
Falls Fahrzeit kürzer als sonst:
  Weg als Lieblingsweg merken
wiederhole bis zufrieden

Optimierung im Alltag

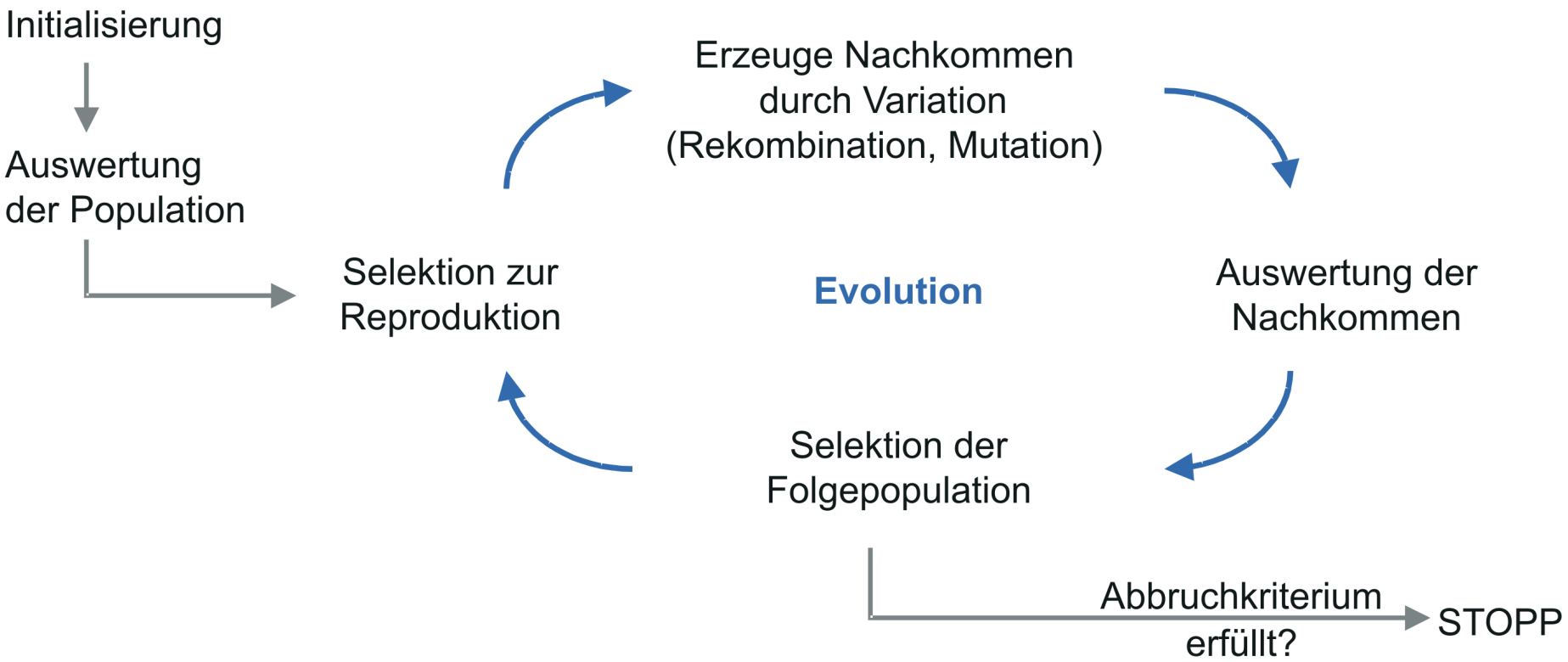
Alltagsproblem:	Optimierproblem:
Schnellster Weg von zu Hause zur Uni?	minimiere Fahrzeit
Irgendeinen Weg ausprobieren.	Initialisierung
Zeit messen.	Funktionsauswertung
	Tue:
Weg leicht verändern	erzeuge Variation
ausprobieren und Zeit messen.	Funktionsauswertung
Falls Fahrzeit kürzer als sonst:	
Weg als Lieblingsweg merken	Selektion
wiederhole bis zufrieden	bis Abbruchkriterium erfüllllt

Dies ist ein evolutionärer Algorithmus!

Evolutionäre Algorithmen (EA)

Inspiriert durch darwinistische Vorstellung der natürlichen Evolution:
sukzessive Verbesserung durch Reproduktion, Variation, Auslese

Ziel ist Optimierung, nicht exakte Nachahmung der Natur!



(Biologische) Fachbegriffe 1

- Genom (Chromosom): Suchpunkt, Lösung $\mathbf{x} = (x_1, \ldots, x_n)$
  Entscheidungsvariable, Objektparameter $x_i, i \in \{1, \ldots n\}$
  Zielfunktionswert $y = f(\mathbf{x})$ des Optimierproblems
  Strategieparameter $\mathbf{w}$: Parameter der Variation
  Fitnesswert z: Zielfunktionswert (in Abhängigkeit der Population)
- Individuum $\mathbf{a} = (\mathbf{w}, \mathbf{x}, y, z)$: Informationen zu einer Lösung
  Population P_t: Multimenge von Individuen in Generation t
- Genotypraum: Suchraum S des EA $G_1 \times \ldots \times G_n, \quad x_i \in G_i$
  Repräsentation: Kodierung des Genotypraums (reellwertig, ganzzahlig, ...)
- Phänotypraum: Suchraum A des Optimierproblems
- $f(\mathbf{x}) = h_2 \circ g \circ h_1$
  Genotyp-Phänotyp-Abbildung: $h_1 : S \longrightarrow A$
  eigentliches Optimierproblem $g : A \longrightarrow B$, B zumindest partiell geordnet
  Abbildung auf vollständig geordneten Raum $h_1 : B \longrightarrow C$

(Biologische) Fachbegriffe 2

- Reproduktion: Generierung von Suchpunkten durch Variation
- Elter: Individuum aus dem ein neues erzeugt wird
  Nachkomme: neues Individuum
- Variation: Rekombination und/oder Mutation
  Mutation: leichte Abwandlung eines Individuums (Elter)
  Rekombination: Vermischung von mehreren Individuen (Eltern)
- Selektion: Auswahl von Individuen
- Generation: eine Iteration (Schleifendurchlauf) des EA

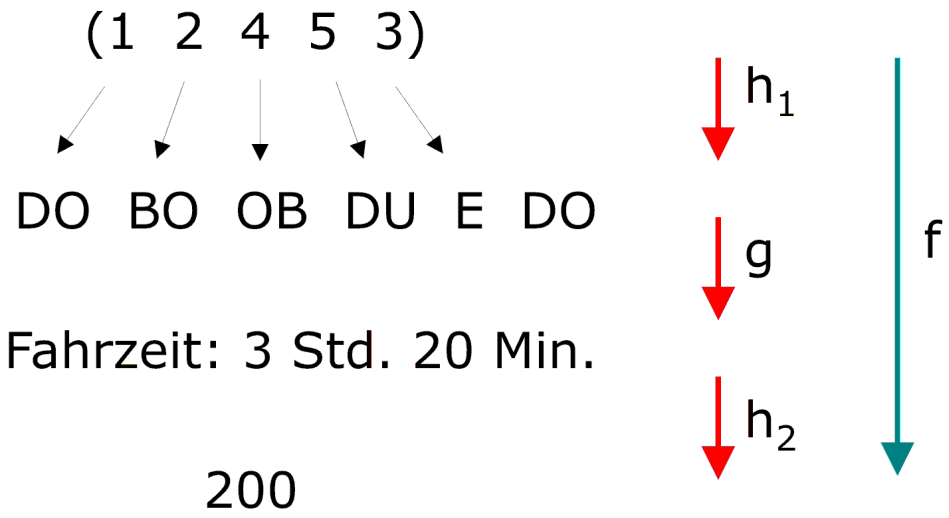
Bei einfachen Szenarien:

Fitnesswert = Zielfunktionswert,
keine Strategieparameter,
$\Rightarrow$Individuum oft synonym gebraucht zu Genom/Suchpunkt/Lösung

Formalisierung am Beispiel TSP

Traveling Salesperson Problem (TSP) ist NP-vollständig
$\Rightarrow$kein effizienter Algorithmus unter Annahme P $\neq$ NP
$\Rightarrow$Anwendung von Heuristik gerechtfertigt

Phänotypraum A: Reihenfolge von n Städten
Genotypraum S: Permutation der Länge n
Zielfunktion: Fahrzeit minimieren
Fitness: Fahrzeit
Mutation auf Permutation: z.B. vertausche benachbarte Elemente



Algorithmische Einordnung von EA

Randomisierte Suchheuristiken zur Black-Box-Optimierung

- Heuristik
  durchdachtes Vorgehen, bei dem aber nicht sicher, ob es zum erwünschten Ergebnis führt
- Suche
  optimale Lösung gesucht im Raum möglicher Lösungen
- randomisiert
  Entscheidungen während des Suchprozesses probabilistisch
- Black-Box-Optimierung
  Algorithmus kennt das zu optimierende Problem nicht
  zu Suchpunkt erhält Algorithmus eine Bewertung (extern)
  Verhalten des Algorithmus abhängig von bisherigen Suchpunkten, Funktionswerten

Anwendungsbereich von EA

- keine geschlossene mathematische Formulierung,
  aber algorithmische Beschreibung
  $\Rightarrow$Funktionsauswertung durch Simulation
- Problem ist unbekannt, schlecht verstanden
  Wie optimiert man ein Problem, das man nicht kennt?
  Steuergrößen verändern und prüfen, ob Ergebnis verbessert
- Problem vielleicht einfach, aber kein Algorithmendesigner zur Verfügung
- Problem nicht schwierig, aber Teilprobleme verändern sich oft
- viel Rechenkapazität vorhanden
- schnell praxistaugliche Lösung benötigt, muss nicht optimal sein

Einfacher Algorithmus: (1+1)EA

$t = 0$	Generationenzähler t
Wähle $\mathbf{x}_0 \in S$ gleichverteilt zufällig	Initialisierung
$y_0 = f(\mathbf{x}_0)$	Funktionsauswertung
Tue	Generationsschleife
$\mathbf{x}' = \text{Mutation}(\mathbf{x}_t)$	Variation: Mutation
$y' = f(\mathbf{x}')$	Funktions-, Fitnessauswertung
Falls $y' \leq y_t$	Selektionskriterium (Minimierung)
$\mathbf{x}_{t+1} = \mathbf{x}'$; $y_{t+1} = y'$	
sonst	Folgepopulation: Lösung $\mathbf{x}_{t+1}$
$\mathbf{x}_{t+1} = \mathbf{x}_t$; $y_{t+1} = y_t$	
$t = t + 1$	Generationenzähler erhöhen
bis Abbruchkriterium erfüllt	Abbruchkriterium

Design der Operatoren eines EA

Im Folgenden Design-Richtlinien und Übersicht populärer Realisierungen der Operatoren von EA

- Initialisierung
- Selektion
- Mutation für veschiedene Repräsentationen
- Rekombination für veschiedene Repräsentationen
- Abbruchkriterium

Initialisierung

Wahl der initialen Werte für Genom, Strategievariablen der Individuen der Startpopulation

- bekannte Lösung (die es zu verbessern gilt)
  + \+ gute Ausgangssituation
  + – eventuell wird wieder ähnliche Lösung gefunden
- zufällig gleichverteilt im Suchraum
  EA unbeeinflusst von Vorwissen

Selektion

Selektion an zwei Stellen in EA:

Selektion zur Reproduktion: wähle Eltern

Selektion der Folgepopulation: wähle Individuen für nächste Generation

Richtlinie:

bei besserer Fitness, keine schlechtere AuswahlWkeit

mehrfache Wahl des gleichen Individuums erlaubt

Uniforme Selektion:

wähle Individuum gleichverteilt aus Population

Fitness-proportionale Selektion:

AuswahlWkeit abhängig von Fitness

Schnitt-Selektion (deterministisch):

wähle k fitness-beste Individuen

Plus-Selektion: wähle aus aktueller Population und Nachkommmen

Komma-Selektion: wähle nur aus Nachkommmen

Turnier-Selektion:

wähle k zu vergleichende Individuen, wähle darunter das fitness-beste

Mutation

Mutation von Genom, Strategieparametern

Maß für Verschiedenheit/Abstand: Metrik auf Suchraum S

Def.: Metrik

Abbildung $d : S \times S \to \mathbb{R}_0^+$ heißt Metrik auf $S :\iff \forall \mathbf{x}, \mathbf{x}', \mathbf{x}'' \in S$ gilt:

$d_S(\mathbf{x}, \mathbf{x}') > 0$ und $d_S(\mathbf{x}, \mathbf{x}') = 0 \iff \mathbf{x} = \mathbf{x}'$ Nichtnegativität

$d_S(\mathbf{x}, \mathbf{x}') = d_S(\mathbf{x}', \mathbf{x})$ Symmetrie

$d_S(\mathbf{x}, \mathbf{x}') \leq d_S(\mathbf{x}, \mathbf{x}'') + d_S(\mathbf{x}'', \mathbf{x}')$ Dreiecksungleichung

Design-Richtlinien für Mutationsoperator $m : S \to S$

- kleine Änderungen wahrscheinlicher als große
  $\forall \mathbf{x}, \mathbf{x}', \mathbf{x}'' \in S : d_S(\mathbf{x}, \mathbf{x}') < d_S(\mathbf{x}, \mathbf{x}'') \Rightarrow Pr(m(\mathbf{x}) = \mathbf{x}') > Pr(m(\mathbf{x}) = \mathbf{x}'')$
  bei kleinerem Abstand (= Metrikwert) zu Elter x, größere ErzeugungsWkeit
- Suche ungerichtet
  $\forall \mathbf{x}, \mathbf{x}', \mathbf{x}'' \in S : d_S(\mathbf{x}, \mathbf{x}') = d_S(\mathbf{x}, \mathbf{x}'') \Rightarrow Pr(m(\mathbf{x}) = \mathbf{x}') = Pr(m(\mathbf{x}) = \mathbf{x}'')$
  alle Suchpunkte mit gleichem Abstand zum Elter x gleiche ErzeugungsWkeit

Mutation für verschiedene Repräsentationen

zunächst: $\mathbf{x}'$ ist Kopie von Elter $\mathbf{x}$

Mutation in $\mathbb{R}^n$ (reellwertige Vektoren)

Addiere Zufallszahl zu x_i'

Zufallszahl Normalverteilt, Cauchy-verteilt

Mutation in $\mathbb{B}^n$ (boolesche Vektoren)

Standard-Bit-Mutation:

invertiere Bit x_i' mit Wkeit p_m

k-Bit-Mutation:

wähle zufällig gleichverteilt k verschiedene zu invertierendes Bits in $\mathbf{x}'$

Mutation in $\mathcal{S}_n$ (Permutationen)

Vertauschung:

wähle zufällig gleichvert. 2 Positionen $i \neq j \in \{1, \ldots, n\}$,

vertausche x_i', x_j'

Sprung:

wähle zufällig gleichverteilt 2 Positionen $i \neq j \in \{1, \ldots, n\}$,

verschiebe x_j' nach Position i, verschiebe Elemente zwischen i, j passend

Skalierung der Mutation

Vorgestellte Operatoren bilden Basis für kompliziertere

Skalierung: Änderung der Stärke der Veränderung

$\mathbb{R}^n$
Zufallszahl multipliziert mit Schrittweite s vor Addition

$\mathbb{B}^n$
Änderung der Mutationswahrscheinlichkeit p_m,
Änderung von k bei k-Bit-Mutation

$\mathcal{S}_n$
k Vertauschungs-, Sprungoperationen

Rekombination

Rekombination von Genom, Strategieparametern

Design-Richtlinien für Rekombinationsoperator $r : S^\rho \rightarrow S$

Nachkomme erzeugt aus ρ Eltern aus $E \subset S^\rho$

- Nachkomme liegt „zwischen“ Eltern
  $\forall \mathbf{x}^{(i)} \in E, i \in \{1, \ldots, \rho\}$:
  $\max\{d_S(r(E), \mathbf{x}^{(i)})\} \leq \max_{\mathbf{x}^{(j)} \in E}\{d_S(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})\}$
  max. Abstand zu jedem Elter $\leq$ Abstand zu dessen weit entferntesten Elter
- kein bestimmter Elter bevorzugt
  $\forall \mathbf{x}^{(i)}, \mathbf{x}^{(j)} \in E, i, j \in \{1, \ldots, \rho\}$:
  $Pr(d_S(r(E), \mathbf{x}^{(i)}) = \alpha) = Pr(d_S(r(E), \mathbf{x}^{(j)}) = \alpha)$
  Wkeit für bestimmten Abstand für alle Eltern gleich

Rekombination

Rekombination in $\mathbb{R}^n$ (reellwertige Vektoren)

Diskrete Rekombination: Elternwert unverändert übernehmen

k-Punkt-Rekombination:

wähle 2 Elter, wähle k versch. Vektor-Positionen,

übernehme Teilstücke alternierend aus Eltern

uniforme Rekombination:

wähle ρ Elter,

übernehme für $\mathbf{x}'_i$ zufällig gleichvert. einen Wert $\mathbf{x}_i^{(j)}$, $j \in \{1, \ldots, \rho\}$

arithmetische Rekombination: Vermischung der Elternwerte

wähle (zufälligen) Gewichtungsvektor $\mathbf{v}$, mit $\sum_{i=1}^{\rho} \mathbf{v}_i = 1$, $\mathbf{v}_i \geq 0$

$\mathbf{x}'_i$ ist gewichtete Summe der Elternwerte an Position i (Konvexkombination)

intermediär: alle Gewichte $1/\rho \rightarrow$ Schwerpunkt der Eltern (determininistisch)

Rekombination in $\mathbb{B}^n$ (boolesche Vektoren)

Diskrete Rekombination: k-Punkt-Rekombination, uniforme Rekombination

Rekombination in $\mathcal{S}_n$ (Permutationen)

Idee: Elemente eines Elter gemäß Ordnung von anderem Elter,
stark problemspezifisch

Abbruchkriterium

EA hat keinen konzeptionellen Endpunkt

Selbst wenn man ein Optimum gefunden hat, kann man das (in der Praxis) meist nicht als solches verifizieren

Abbruch bei benutzerdefinierter Bedingung

- Beschränkung von Ressourcen:
  Anzahl Funktionsauswertungen
  Anzahl Generationen
  CPU-Zeit, Realzeit
- Abhängig von (relativer) Qualität:
  Wurde ein bestimmter Wert erreicht?
  Bewirkte die letzte Generation keine große Verbesserung?
- Ungeduld

Zusammenfassung

Themen der heutigen Vorlesung:

- Was ist ein Optimierproblem?
- Welche Optimierprobleme sind schwierig?
- Was ist ein evolutionärer Algorithmus (EA)?
- Fachbegriffe
- Anwendungsgebiet
- Algorithmische Einordnung
- Beispiel: (1+1)EA
- Design-Richtlinien für Operatoren
- Beispiele üblicher Operatoren

Literatur

- Optimierung
  Vorlesung „Modellgestützte Analyse und Optimierung“ von Peter Buchholz
- EA
  Vorlesungen von Thomas Jansen (in diesem Semester),
  Ingo Wegener, Hans-Georg Beyer