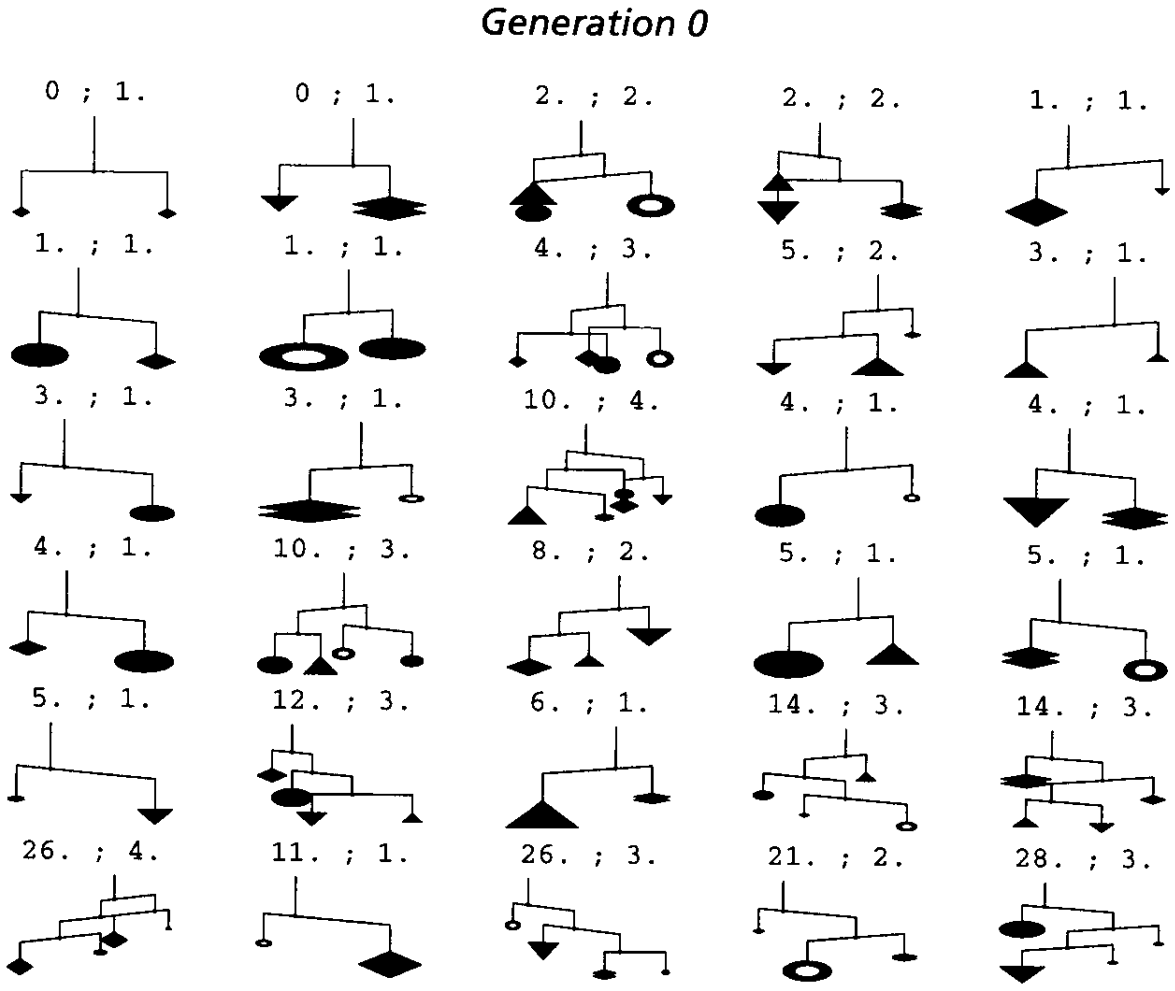
weitere Beispiele für Anwendungen evolutionärer Algorithmen:

Evolution ausbalancierter Mobiles
(Jacob 1997)

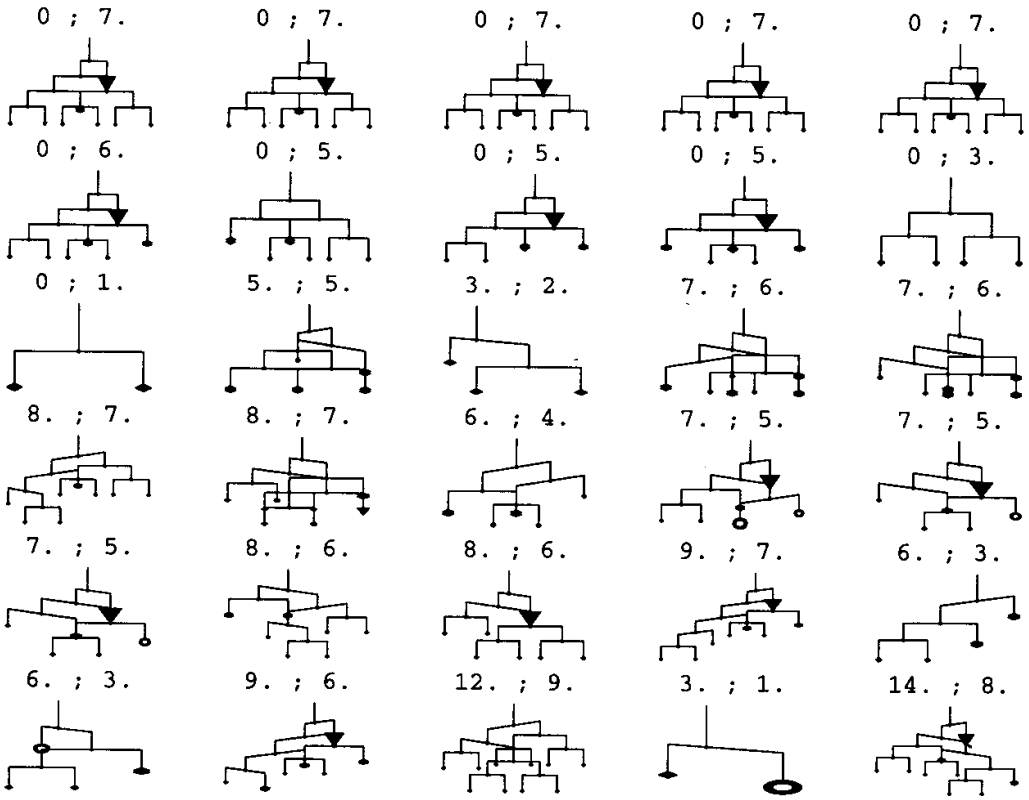
Genetisches Programmieren: Mobiles durch Terme (Bäume) repräsentiert

Fitness: Annäherung der Gleichgewichtsbedingung

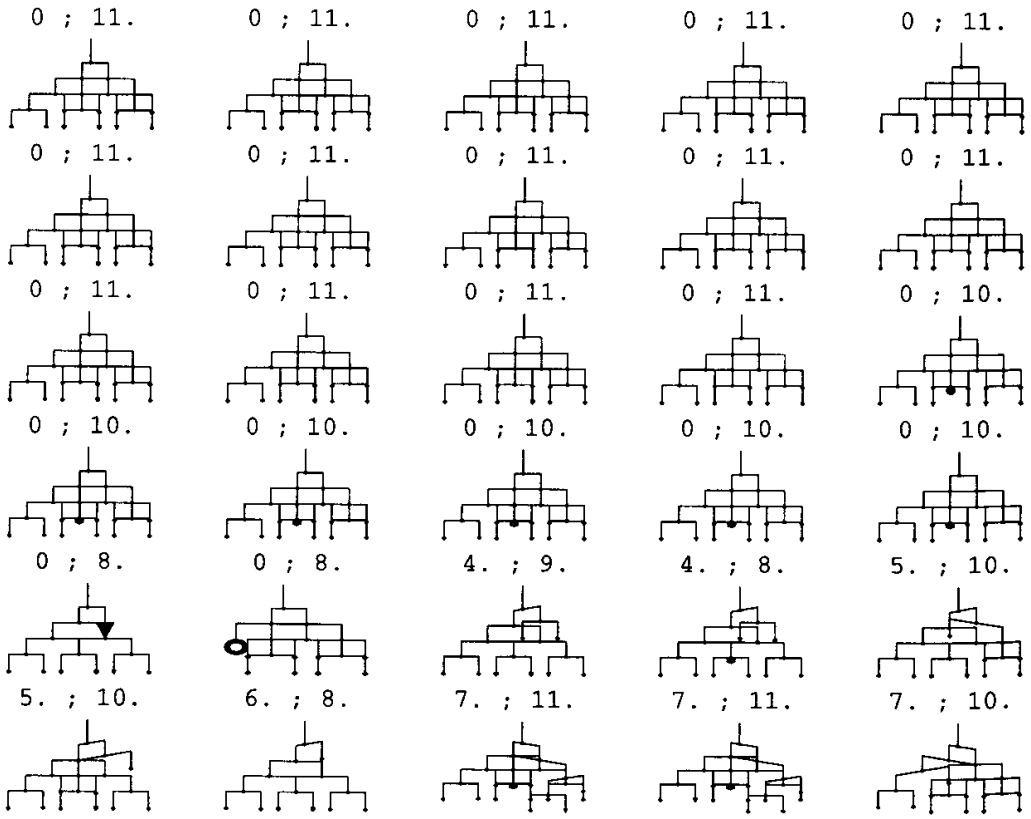


(Abbildungen aus Jacob 1997)

Generation 20



Generation 50



Evolution von Formelausdrücken zur Approximation von Zahlenfolgen, deren Anfangsstück vorgegeben ist
(Neuse 2002)

- typisches KI-Problem
- die 200 ersten Werte der Folge werden vorgegeben
- Operatoren:
  0-stellig: ganzzahlige Konstanten 0 bis 255; *e* und π
  1-stellig: –*x*, 1/*x*, exp, log, sqrt, Kopierbefehl
  2-stellig: +, –, *, /, Potenz
- unmittelbare und adressbezogene Interpretation von Speicherinhalten möglich (vgl. Core Wars)
- 16 Zellen für Zwischenergebnisse
- move- und copy-Bits zum "Umschreiben" des Programms
- fester Speicherbereich von 64 Drei-Byte-Befehlen (diese 64 Felder ergeben zusammen 1 Individuum) – in einfacheren Versionen auch 16 oder 32 Befehle
- binäre Codierung
- in jeder Generation Überprüfung der Übereinstimmung von Formelergebnis und vorgegebener Folge nur anhand von 5 zufällig gewählten Beispiel-Zahlen (aus den 200 vorgegebenen; Rechenzeitgründe!) – für alle Individuen einer Generation dieselben
- GA mit Populationsgröße 120
- Fitness: Minimum der Quotienten berechnete Zahl / vorgegebene Zahl und der inversen Quotienten
- Selektion: Kombination von Truncation (nur die bessere Hälfte der Population überlebt) und roulette-wheel-Selektion
- Mutation (bitweise) und Crossing over; Mutationsrate zwischen 10% und 30%
- Implementation auf PC

Beispiel: Fibonacci-Zahlen F_n:
1; 1; 2; 3; 5; 8; 13; 21; ... ; $2{,}80571173 \times 10^{41}$

- mögliches (ideales) Ergebnis wäre die Binet'sche Formel gewesen:

$$F_n = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$$

- der GA war (bisher) nicht erfolgreich im Finden einer korrekten, exakten Formel (die extrapolationsfähig ist)
- bisher bestes Ergebnis nach ca. 40 000 Generationen:

$$f(x) = \frac{A(x)}{B(x)C(x)} \quad \text{mit}$$

$$A(x) = \sqrt{e^x} \cdot \pi^{\sqrt{x/71}/(x+\sqrt{x})}$$

$$B(x) = \frac{x}{71} + e^{\frac{x}{71}} + \frac{\left(\sqrt{\log x}\right)^{x/71}}{\pi}$$

$$C(x) = \sqrt{e^{x/71}} + \pi^{D(x)}$$

$$D(x) = \frac{\sqrt{\frac{x}{71}}\left(\frac{x}{71} \cdot e^{\frac{x}{71}-1} - x - \sqrt{x}\right)}{x+\sqrt{x}}$$

Abweichungen $f(n)/F_n$:
liegen zwischen 0,99 und 1,015 im gesamten Bereich
von n=8 bis n=200 (Fitness bei 98,6%)

aber für n=250: 0,943, n=400: 0,632, n=500: ca. 10^{-5} !

- d.h. die gefundene Approximationslösung gilt nicht mehr in einem Bereich, der über den während des GA getesteten Bereich (1–200) hinausgeht

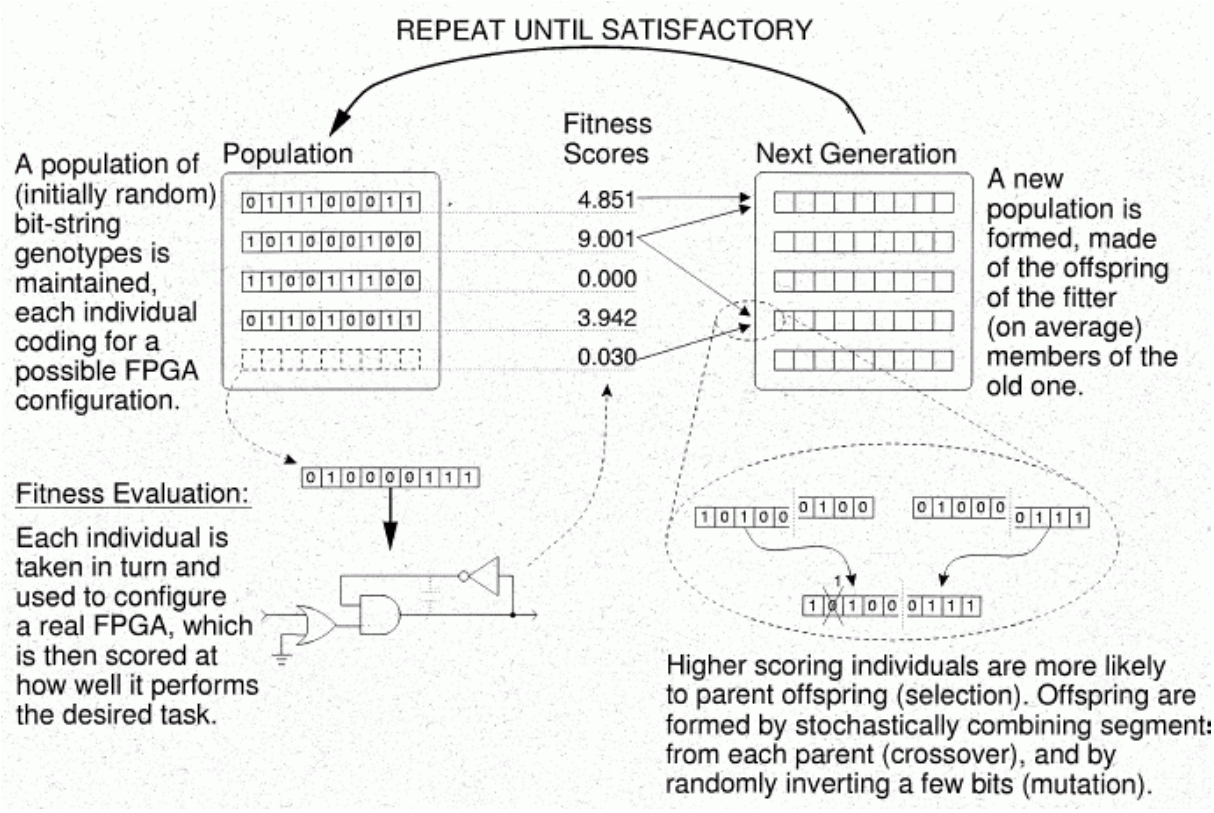
- wenn man *e* und π als Konstanten ausschließt, ist das Ergebnis (bisher) schlechter!

Evolution von Hardware (Adrian Thompson, zit. nach Pfeifer et al. 2002)

Konfiguration von FPGAs (*Field Programmable Gate Arrays*):

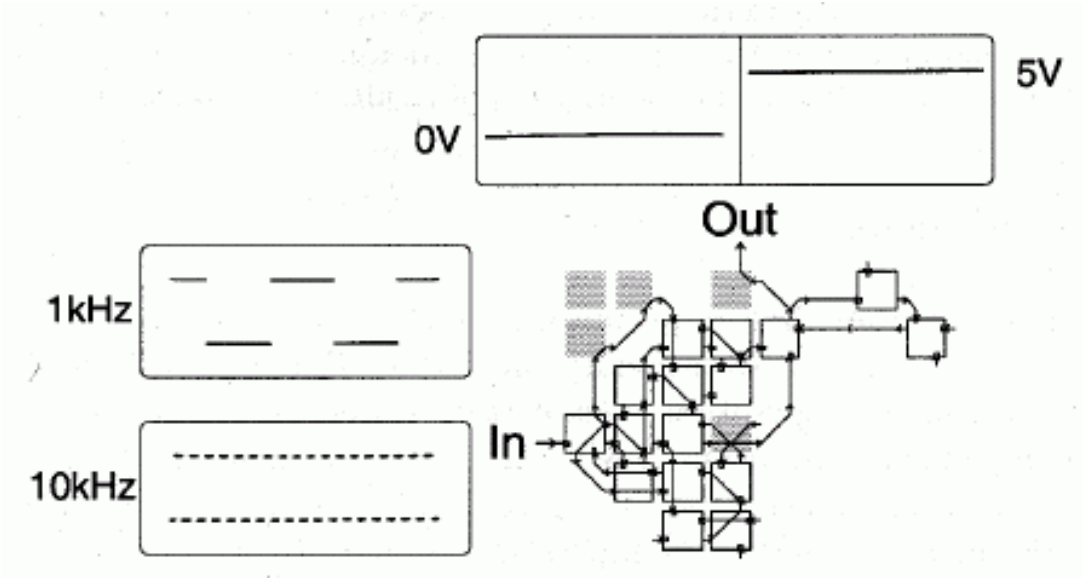
- VLSI-Chip mit vorgefertigten Komponenten, die durch konfigurierbare Schalter verbunden sind
- Schalterstellungen sind in "configuration memory" gespeichert
- dieser Speicher kann über angeschlossenen Rechner manipuliert werden
- jede Schalterstellung erzeugt einen neuen realen physikalischen Schaltkreis, dessen Eigenschaften mit Test-Software geprüft werden können (Fitness-Auswertung)

GA-Anwendung:



in Thompson's Beispiel war die Aufgabe des Schaltkreises, 1kHz- und 10kHz-Audiosignale (als Input) zu unterscheiden (Mustererkennung).

- Fitnessfunktion: Differenz der Ausgabespannungen bei diesen beiden Input-Signalen.
- keine synchronisierende Uhr eingesetzt!
- beliebige Eigenschaften der Silizium-Schaltkreise können ausgenutzt werden



Ergebnisse:

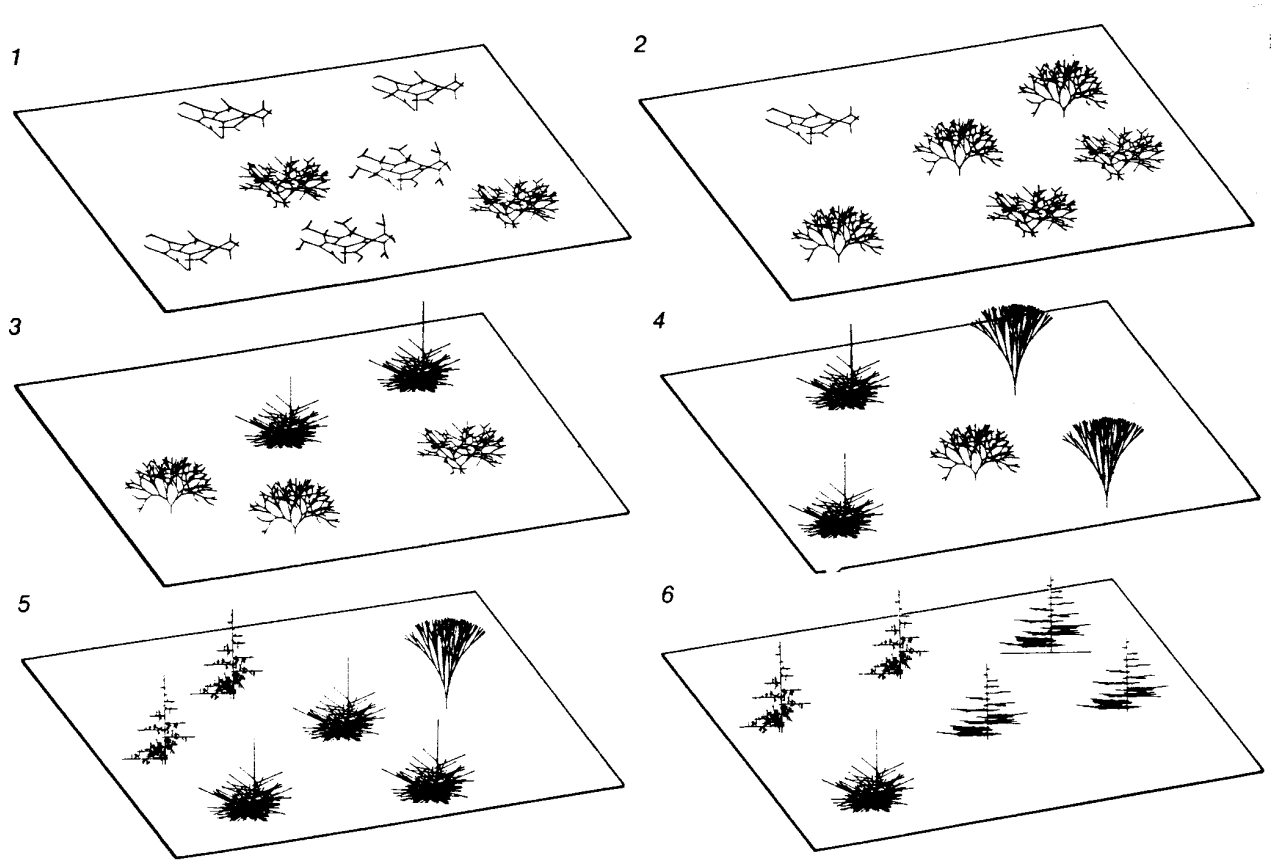
- der GA ist erfolgreich
- perfektes Verhalten wird schon mit einem 10×10-Array aus dem insgesamt zur Verfügung stehenden 64×64-Array erreicht; konventioneller Schaltungsentwurf würde einen größeren Bereich beanspruchen
- innerhalb des 10×10-Bereichs werden nicht alle Zellen benutzt; einige werden nur "indirekt" benutzt (ohne direkte Verbindung: graue Zellen in der Abb.) – müssen auf andere Weise mit dem Rest der Schaltung interagieren (elektromagnet. Kopplung)

- somit "unerwartete" Lösung; deutliche Unterschiede zu konventionellem Schaltungsentwurf

Anwendung von GA auf Morphogenese (vgl. Kap. 2)

Evolution der Landpflanzen (Niklas; vgl. oben)

selbes multikriterielles Optimierungsproblem, jetzt mittels GA gelöst (hier für Optimierung der Lichtinterzeption und Sporen-ausbreitung):



im Prinzip erhält man dieselben "adaptive walks" wie bei der direkten "hill climbing"-Methode

– bei niedrigdimensionalen Genomräumen und (relativ) "einfachen" Fitnesslandschaften bietet der GA keine Vorteile gegenüber konventionellen Optimierungsmethoden

Evolution von L-Systemen

bietet sich an, da schon symbolische Repräsentation der Morphologie vorhanden
Regelsystem = Genotyp,
3D-Struktur (nach Turtle-Interpretation) = Phänotyp

Jacob 1997: ArtFlowers-System (mit Mathematica realisiert)

- Term-Repräsentation der L-Systeme

- Die L-Systeme sind die Individuen, die zugehörigen Phänotypen entstehen durch Anwendung von *N* Schritten und anschließender Turtle-Interpretation (nach jedem GA-Schritt für alle Individuen der Population!)

- Evolutionsoperatoren können sowohl komplette Regeln, als auch Sequenzen oder geklammerte Module innerhalb der rechten Regelseiten verändern

- mit Hilfe von "Schablonen" wird sichergestellt, dass nur Terme selektiert werden, für die die betreffenden Operatoren auch sinnvoll sind

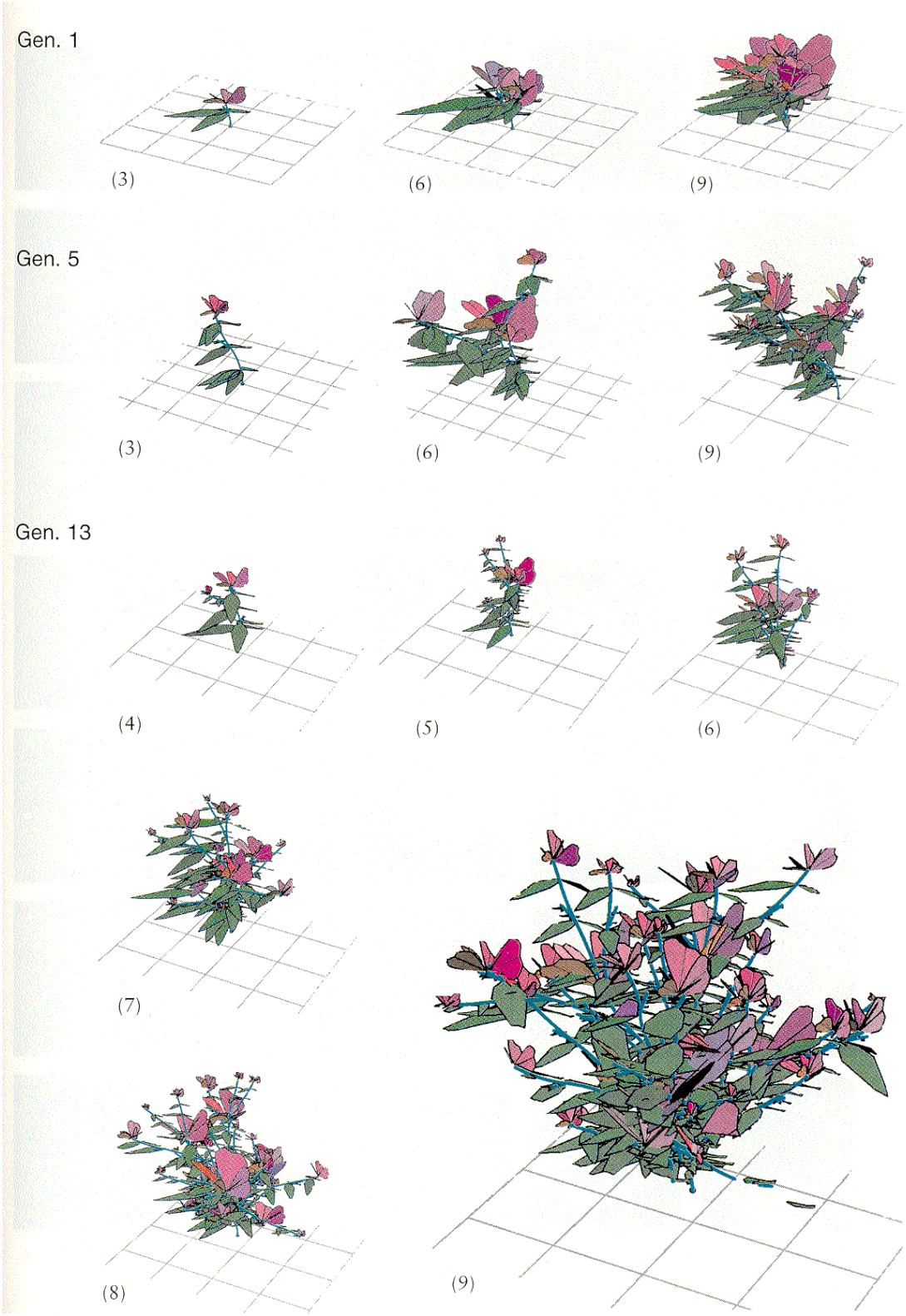
- die Schablonen sind mit Gewichten versehen, so dass z.B. die Veränderung einer kompletten Regel seltener ist als die von Teilausdrücken der rechten Regelseite

- Evolutionsoperatoren: Mutation, crossing over, Deletion, Duplikation, Permutation

- Fitness (beispielsweise): Produkt aus Anzahl der Blüten und Volumen der bounding box der Pflanze nach vorgegebener Schrittzahl

Beispiel: Wachstumsschritte einiger Pflanzenstrukturen aus evolvierten L-Systemen

("Gen." bezeichnet die Generationszählung des GA, die eingeklammerte Ziffer die Zahl der abgearbeiteten Schritte des L-Systems)



(aus Jacob 1997)

Evolution von zellulären Automaten
(Mitchell 1996)

Einfache Codierung als String: Output der Transitionsfunktion des CA wird für alle möglichen Inputkombinationen (z.B. in lexikografischer Ordnung) hintereinandergeschrieben

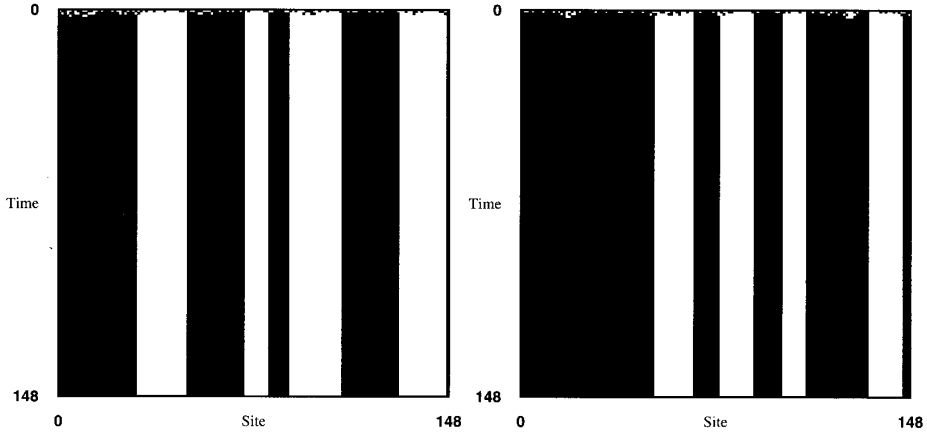
Nichttriviale Aufgabenstellung:
es sollen 1-dim. binäre CA (mit festem Radius *r*; auf ringförmigem Band fester, ungerader Länge) gefunden werden, die zu einer gegebenen Zahl *M* folgendes leisten:

- wenn die durchschnittliche Dichte ρ der Einsen in der Anfangskonfiguration kleiner als ½ ist, liefert der CA nach *M* Schritten ein Band, was mit lauter Nullen besetzt ist
- wenn $\rho > ½$ ist, ein Band mit lauter Einsen

("$\rho_c = ½$ - Aufgabe"; ρ_c = "kritische Dichte")

Beachte: die Aufgabe verlangt Informationsfluss über "große" Distanzen (im Vergleich zum Radius *r* der Nachbarschaft), da die Dichte der Einsen in der Anfangskonfiguration großräumig unausgewogen sein kann.

Eine einfache "Mehrheitsregel" (Ergebnis der Transitionsfunktion = 1, wenn die Mehrheit der Zellen in der Nachbarschaft Wert 1 hat, sonst 0) erfüllt die Aufgabe *nicht*:



Verhalten eines 1-dim. CA mit Mehrheitsregel, r = 3, Zeitverlauf von oben nach unten. Linke Startkonfiguration mit $\rho < ½$, rechte mit $\rho > ½$.

GA-Anwendung:

- Bandlänge $N = 149$
- Populationsgröße 100
- Initialisierung: 100 zufällig gewählte CA-Regeln
- jedes Individuum (CA-Regel) wird an 100 Startkonfigurationen getestet (Test *aller* möglichen 2^{149} Startkonfigurationen wäre unmöglich!)
- die Startkonfigurationen werden zufällig bestimmt, aber mit einem Algorithmus, der uniforme Verteilung der Einsen-Dichte im Intervall [0; 1] sicherstellt (völlig zufällige Auswahl der Startkonfigurationen würde "zu schwere" Testfälle liefern, da die meisten dann $\rho \approx ½$ hätten).
- $M = 2N$
- Fitness einer Regel: Anteil der 100 Startkonfigurationen, für die sie das korrekte Endmuster liefert ("teilweise richtige" Lösungen werden nicht mitgezählt)
- Selektion: "Elite-Auswahl" (beste 20%) + crossover aus der Elite (80%) mit anschließender doppelter Mutation
- GA mit 100 Generationen
- Bestimmung der Fitness ist rechenintensivster Teil
- 300 GA-Läufe mit unterschiedlichen Zufallszahlen

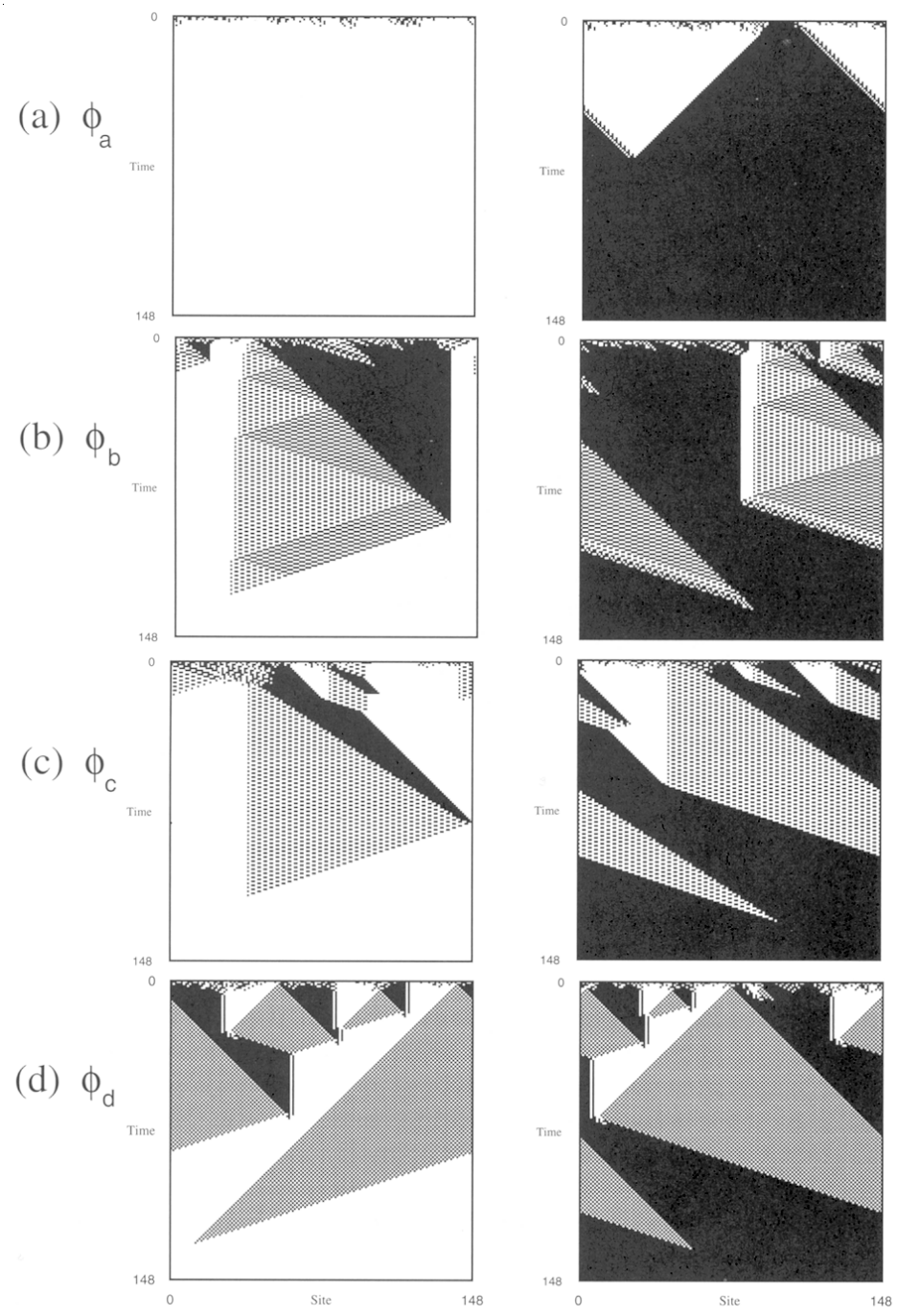
Ergebnisse:
meistens evolvieren nichttriviale, aber relativ unkomplizierte Regeln
z.B. ϕ_a : liefere Band mit Nullen, wenn nicht ein genügend langer Block aufeinanderfolgender Einsen in der Startkonfiguration vorliegt. Ein solcher wird expandiert und liefert ein Band mit nur Einsen. (Fitness $\approx 0{,}9$)
– funktioniert "meistens", weil bei hoher Einsen-Dichte meistens auch ein genügend langer Block auftritt (aber dies ist nicht garantiert).

- diese "block-expanding"-Strategien waren für die Autoren des GA überraschend
- führen aber nur lokale Berechnungen durch, kein Fern-Transfer von Information, der erhofft war
- Fitness wird schlecht bei Einsen-Dichte nah bei 0,5

in einigen Läufen evolvierten bessere Regeln mit "komplexeren" Strategien

- "Partikel-basierte" Regeln mit Signalweiterleitung über größere Distanzen
- keine Regel schafft eine Fitness von 100%

4 Beispiele für die Performance evolvierter CA-Regeln (linkes Diagramm startet jeweils mit $\rho < ½$, rechtes mit $\rho > ½$) (das graue Gebiet in (d) ist ein Schachbrettmuster aus Nullen und Einsen) – die Fitness wächst von (a) bis (d):

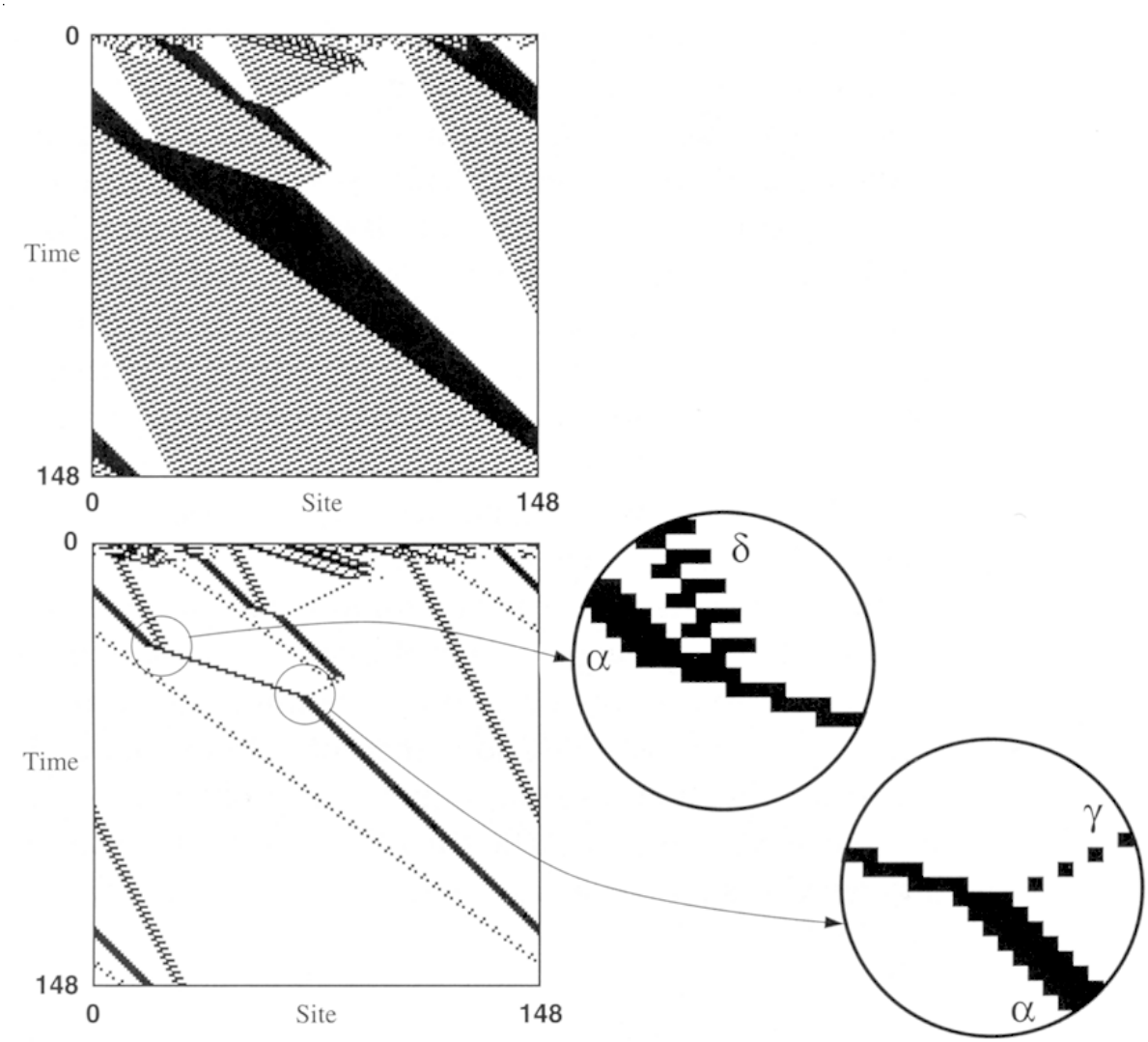


Analyse solcher CA-Regeln:

"Computational mechanics" (Hanson & Crutchfield), betrachtet "reguläre Bereiche" (mit Mustern aus Wörtern einer regulären Sprache), deren Grenzen = "Partikel" = Informationsträger, Kollisionen zwischen Partikeln bewirken Informationsverarbeitung

- Partikel und ihre Interaktionen als "high-level"-Sprache zur Beschreibung von Berechnungen in räumlich ausgedehnten, homogenen Systemen wie CAs.

Analyse eines evolvierten CA durch Herausfiltern aller regulären Bereiche und Betrachtung der Partikel:



Karl Sims' Virtual Creatures
(SIGGRAPH 1994 und Artificial Life IV - Konferenz 1994)

Ziel:
Es sollen 3D-Individuen entwickelt werden, die in einer virtuellen Welt mit simulierter Physik (Gravitation, Reibung, Kollisionserkennung, Flüssigkeitsdynamik...) spezielle Aufgaben besonders gut bewältigen können

- Struktur (Morphogenese) und Verhalten (Regeln, Kontroll-strukturen: neuronales Netz) sind durch den genetischen Algorithmus variierbar

Kodierung der Kreaturen:Gestalt

Genotyp: Strukturbeschreibung
Phänotyp: Ausprägung

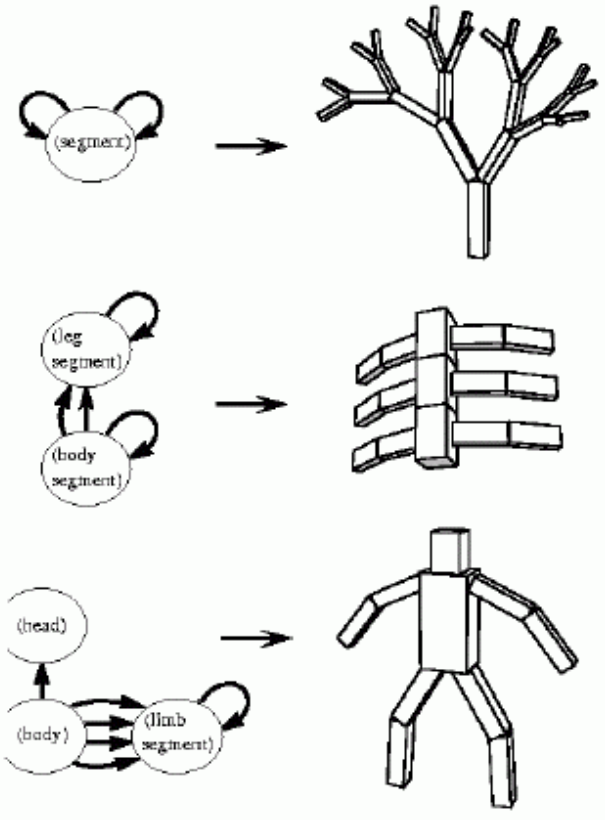
Die Erbinformationen über die Gestalt der Lebewesen wird als *gerichteter Graph* gespeichert.

Die **Knoten** entsprechen je einem **Körperteil** und enthalten folgende Informationen:

- Größe
- Gelenktyp
- Rekursionslimit
- Neuronen
- Verbindungen

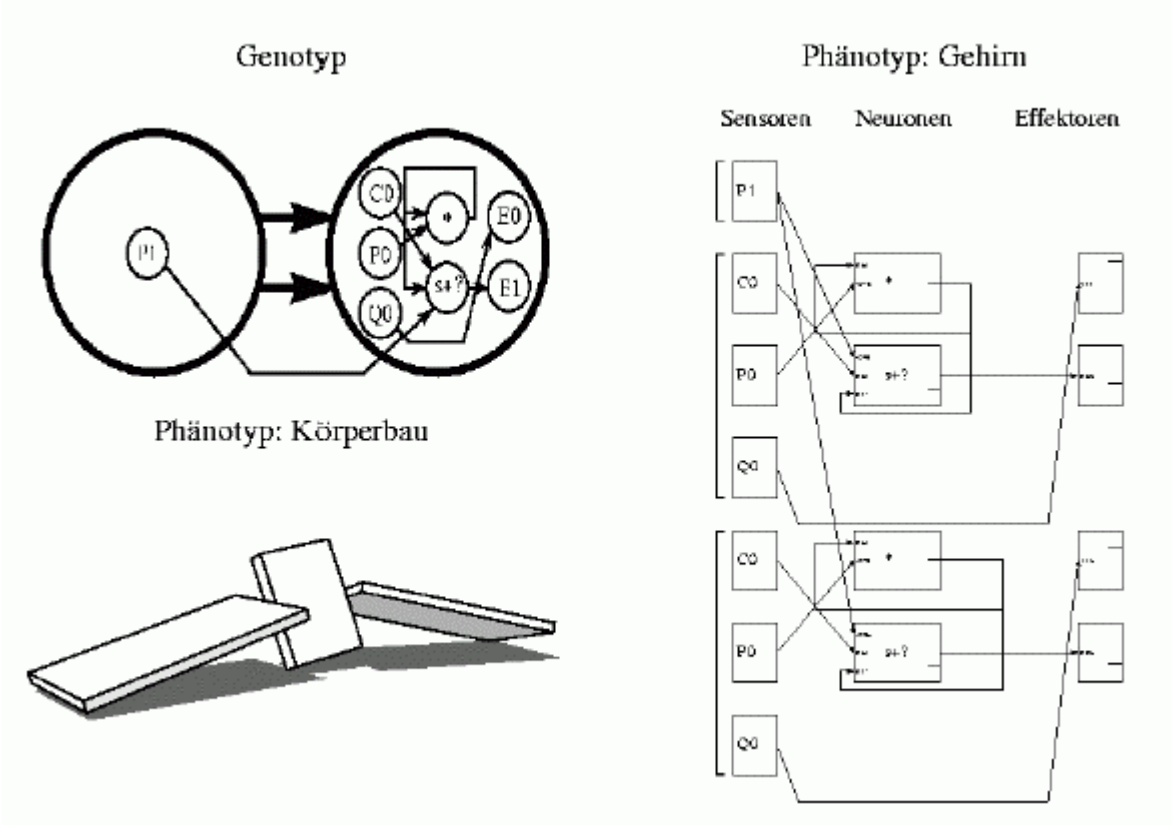
Die **Kanten** entsprechen den **Gelenken** und enthalten folgende Informationen:

- Position
- Orientierung
- Skalierungsfaktor
- Nur-Ende-Flag



Aufbau einer Kreatur:

- segmentiertes System mit Gelenken
- Winkelsensoren und Effektoren für jedes Gelenk
- neuronale Schaltkreise in jedem Segment + "Gehirn" (neuronales Netz)



Codierung durch Graphen:

- "äußerer Graph": Anordnung der Segmente (Glieder) und Gelenke
- "innerer Graph": Verschaltung der Sensoren, Neuronen und Effektoren

Genotyp enthält auch Informationen über Sensoren, Effektoren und Neuronen

Evolution

Ein Genotyp wird in folgenden Schritten mutiert:

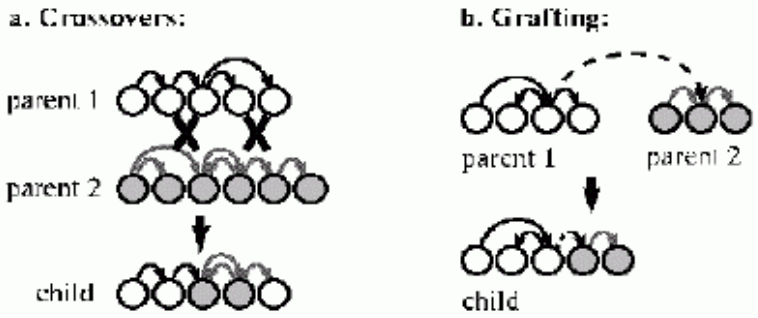
1. Interne Parameter der Knoten werden z. T. verändert.
2. Ein neuer Knoten wird hinzugefügt.
3. Parameter der Kanten werden z. T. verändert; evtl. zeigt die Kante anschließend auf einen anderen Knoten.
4. Neue Kanten werden evtl. hinzugefügt.
5. Nicht (mehr) befestigte Knoten werden entfernt

Das Verfahren besteht aus zwei Phasen:

Zuerst wird der äußere Graph bearbeitet, dann der innere.

Methoden:

1. [40%] Kopieren des Genotyps; dabei treten **Mutationen** auf
2. [30%] **Crossover** (evtl. mit Mutationen)
3. [30%] **Grafting** (evtl. mit Mutationen)



Verwendung von gerichteten Graphen erfordert spezielle Verfahren für "Crossover" (Variante: "Grafting" = Pfropfung)

- Selektion: "truncation", d.h. nur die besten 20 % einer Generation überleben für die Reproduktion
- zusätzlich ist die Anzahl der Nachkommen proportional zur Fitness für die spezifische, vorgegebene Aufgabe

Der beschriebene genetische Algorithmus wurde angewendet, um Lebewesen zu generieren, die eine der folgende Aufgaben besonders gut beherrschen:

1. **Schwimmen**
2. **Fortbewegen an Land**
3. **Springen**
4. **Folgen** einer Lichtquelle
5. Sich im **Wettkampf** gegen ein anderes Individuum durchsetzen

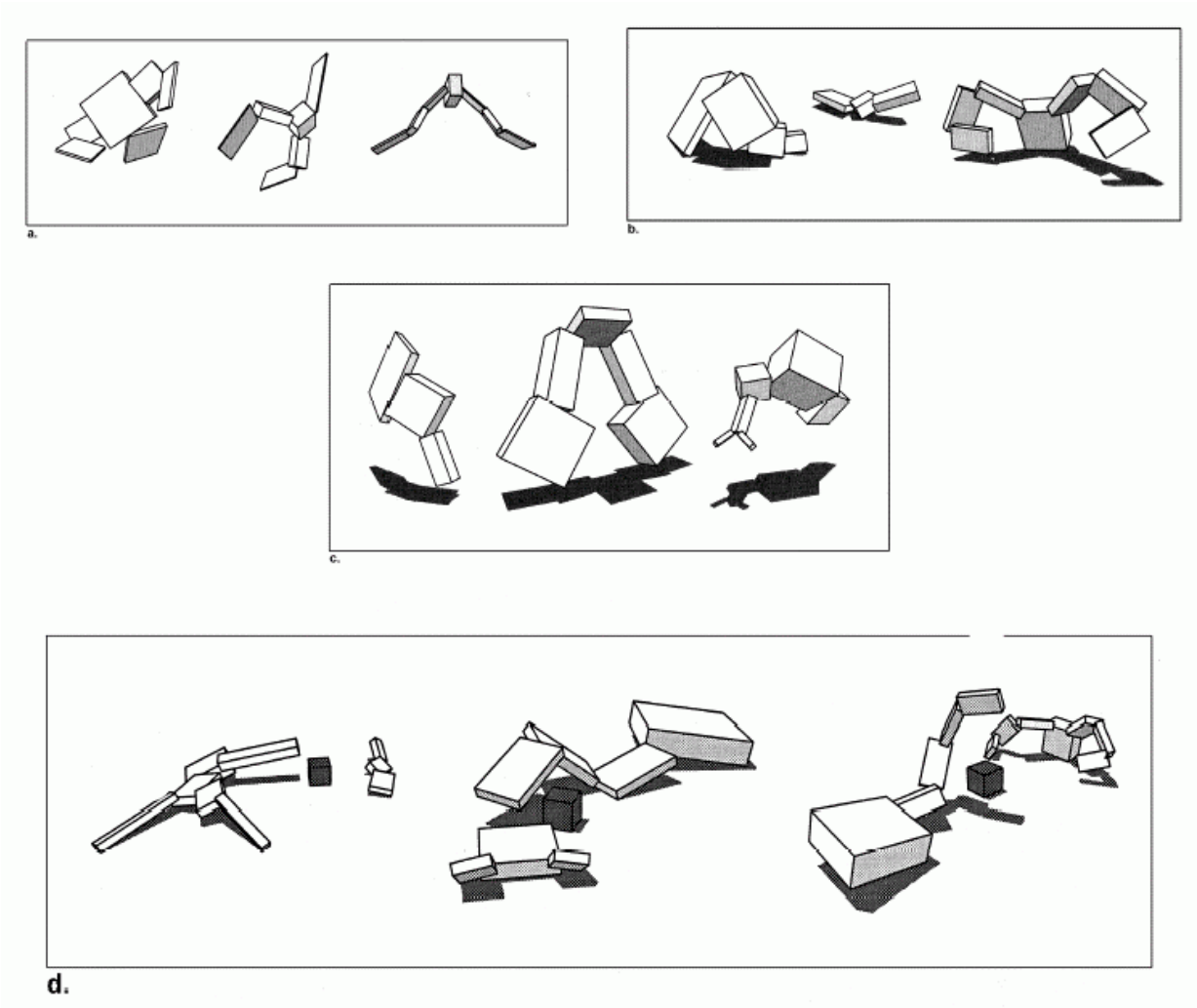
Zur Implementierung:

- Populationsgröße: 300
- jeweils 50 - 100 Generationen pro Evolution
- jeweils max. 10 sec. pro Individuum und Generation zur Qualitätsbewertung
- **Laufzeit**: 3 - 4 Std. pro Evolution mit 100 Generationen auf einer CM-5 mit 32 Prozessoren

weitere Aufgabe: Kontrolle über einen Würfel gewinnen (und um diesen mit anderen Kreaturen konkurrieren) – Fitnessfunktion: Nähe zum Würfel am Ende einer festgelegten Zeitspanne

Ergebnisse:

oben links: Aufgabe "Schwimmen"
oben rechts: Aufgabe "Laufen"
Mitte: Aufgabe "Springen"
unten: Aufgabe "Um Kontrolle über Würfel konkurrieren"



Beobachtungen:

- auch hier "Überraschungseffekte", innovative Lösungen (z.B. Fortbewegung durch Sich-überschlagen)
- darunter auch Lösungen, die in der Natur nicht vorkommen
- hohe Rechenzeit-Anforderungen aufgrund von sehr großem Genom-Raum
- je mehr Einschränkungen man auferlegt, desto weniger Rechenzeit ist nötig, aber auch desto weniger Innovation

das Grundprinzip von Sims' "Creatures" wurde später in einem gleichnamigen Computerspiel weiterverarbeitet.

Stephen (Steve) Grand 1996, "Creatures" (Cyberlife Inc.)

http://www.creatures.co.uk/

Genetischer Algorithmus mit *intrinsischer Adaptation*

(der Fitnesswert eines Genoms ist nicht schon durch das Genom eindeutig festgelegt, sondern hängt auch von der übrigen Population und von räumlichen und zeitlichen Interaktionen zwischen den Individuen ab):

LindEvol (Jan Kim 1996)

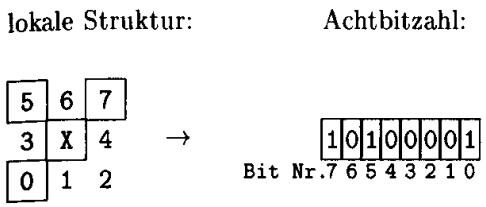
ALife-Experimente zur Simulation der Evolution von Pflanzen – verschiedene Varianten

Grundideen:

- physikalische Komponenten des Modells: Raum (= 2-dim. oder 3-dim. Gitter), Zeit (diskrete Schritte), Energie ("Photonen", kommen von oben; Aktionen einer Pflanzenzelle nur nach Absorption eines Photons möglich)
- genetische Komponenten: Gene, Genome, Genregulation, Mutationen (kein crossing over)
- ökologische Komponenten: Interaktion zwischen Individuen, Konkurrenz um Ressourcen "Energie" und "Raum"

2-dim. Struktur: endlich ausgedehntes, zylindrisch geschlossenes Gitter ("Welt" mit Boden und Decke) mit quadratischen Zellen, Moore-Nachbarschaft

- eine Pflanzenzelle besetzt genau eine Gitterposition, und eine Gitterposition kann von höchstens einer Pflanzenzelle besetzt sein
- einem Pflanzen-Individuum sind endlich viele Pflanzenzellen zugeordnet
- interne Parameter einer Pflanzenzelle: Energieparameter 0 oder 1, bei einigen Varianten zusätzlich 16- oder 32-bit interner Zustand
- lokale Struktur einer Zelle: Belegung der Nachbarzellen (Codierung von 2-dim. lok. Strukturen in 8-bit-Zahlen)



Pflanzenzellen können *Aktionen* ausführen, von denen die meisten Energie verbrauchen (d.h. nur nach Absorption eines Photons möglich sind):

- Generierung einer energielosen Nachbarzelle (Index der zu besetzenden Nachbarzelle kann durch Parameter spezifiziert werden)
- "flyingseed": neue Pflanze (aus einzelner, energieloser Zelle bestehend) wird an einer zufällig ausgewählten freien Gitterposition am Boden der "Welt" generiert
- "localseed": neue Pflanze wird senkrecht unter der Mutterpflanzenzelle generiert
- Exponent zur Bestimmung der Mutationsrate wird um 1 erhöht oder vermindert
- einzelnes Bit des internen Zellzustands für den nächsten Zeitschritt wird gesetzt (verbraucht keine Energie)

Photonen wandern vertikal abwärts und werden von Pflanzenzellen mit 50% Wahrscheinlichkeit absorbiert
$\Rightarrow$ Lichteinstrahlung ist die Energiequelle, die alle "Lebensvorgänge" antreibt

Eine *Pflanze* enthält neben ihren Zellen

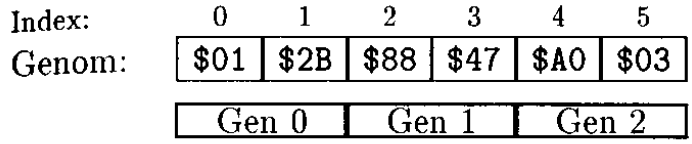
- ein *Genom*: eine Menge von Regeln "Zustand $\rightarrow$ Aktion", die als Binärstrings codiert werden (am Anfang des Simulationslaufs wird Menge von Genomen zufallsgeneriert)
- einen Mutationsexponenten *a* (Mutationsrate = Basisrate des Mutationstyps $\times$ m^a, *m* = Konstante)
- Information über Gesamtzahl ihrer Zellen und Gesamtzahl ihrer energiereichen Zellen ("Gesamtenergie" der Pflanze)

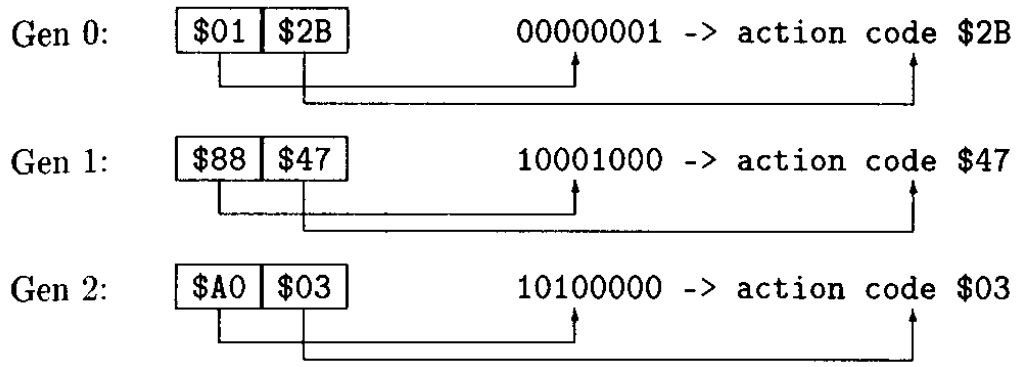
Eine im Genom enthaltene Regel (ein "Gen") wird *aktiviert*, wenn der Zustand einer Zelle (einschließlich der Nachbarschaftssituation) zur linken Regelseite "passt"

- für energieverbrauchende Aktionen muss die Zelle zusätzlich gerade ein Photon absorbiert haben
- von mehreren aktivierten Genen für energieverbrauchende Aktionen wird nur das erste im Genom ausgeführt

⇒ durch Generierung einer Nachbarzelle können bei anderen Zellen und bei der Mutterzelle selbst Gene "ein-" und "ausgeschaltet" werden, da sich die lokale Umgebung ändert (es entsteht ein regulatorisches Netzwerk)

Blockorientierte Genominterpretation:
jedes Gen besteht aus 2 aufeinanderfolgenden Bytes im Genom, die linke und rechte Regelseite codieren



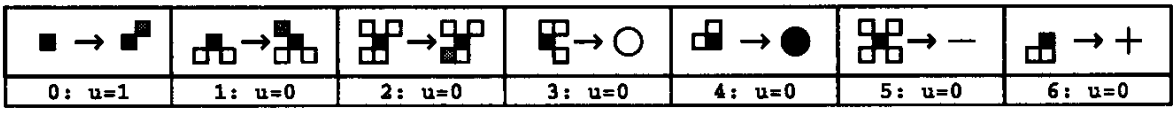


Darstellung eines Genoms (Beispiel):

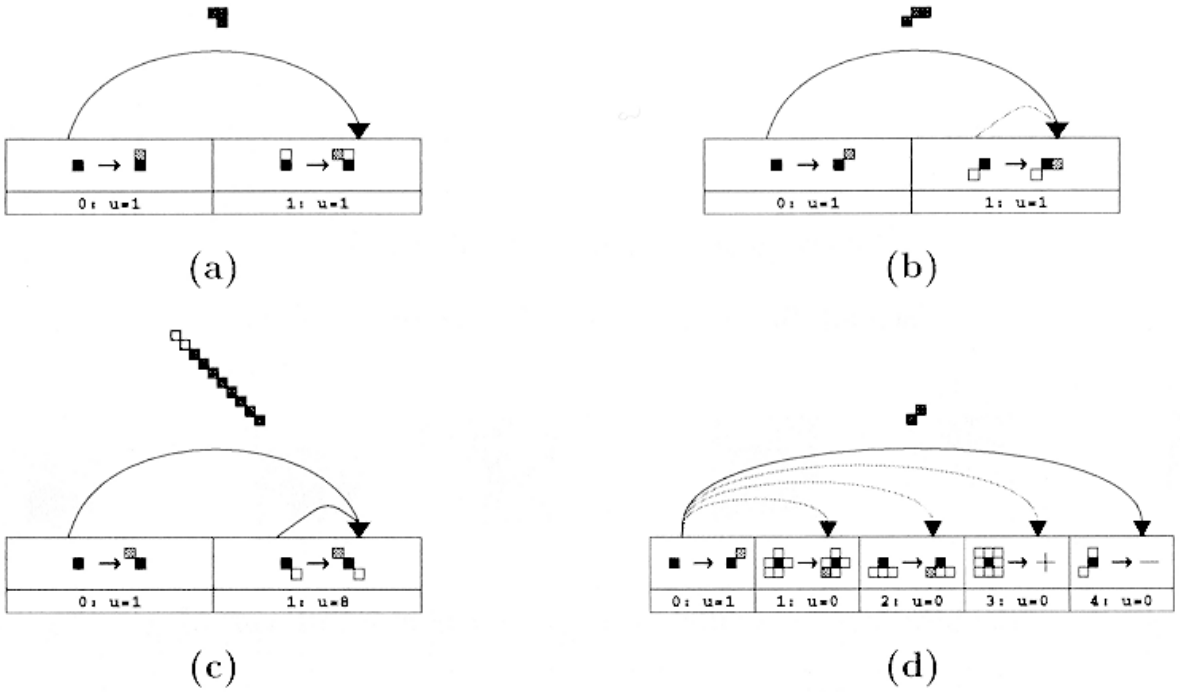
***Listing* des Genoms:**

```
0 [    1]: 00000000 -> divide 7              00 07
1 [    0]: 00000101 -> divide 5              05 05
2 [    0]: 10100011 -> divide 0              a3 b8
3 [    0]: 11000100 -> flying seed           c4 c0
4 [    0]: 01001000 -> local seed            48 d7
5 [    0]: 10100101 -> mut-                  a5 eb
6 [    0]: 00000011 -> mut+                  03 f4
```

Graphische Darstellung des Genoms:



Beispiele für Pflanzen mit blockorientierter Genominterpretation, mit ihren regulatorischen Netzwerken:



(energiereiche Zellen sind schwarz gezeichnet)

Mutationsoperatoren:

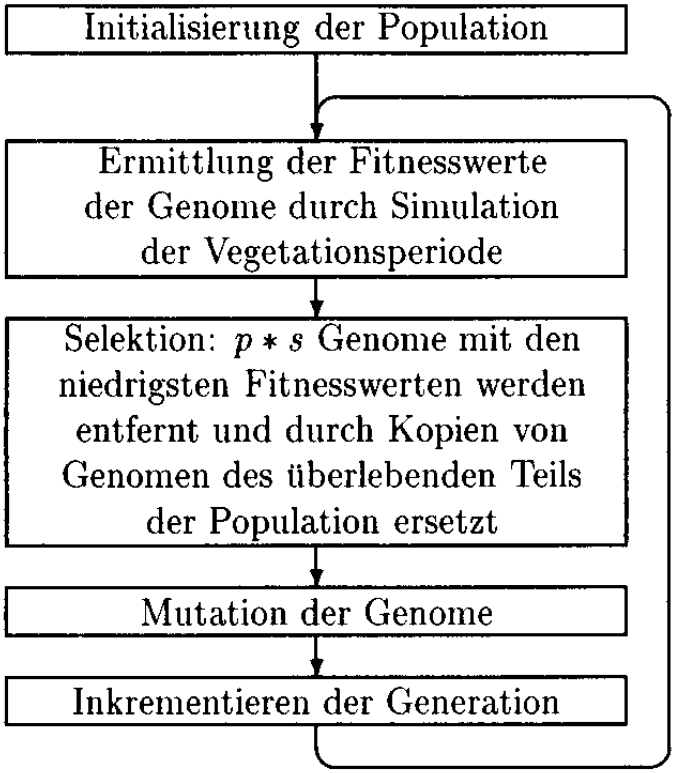
- Änderung (byteweise oder bitweise)
- Insertion
- Deletion
- Genduplikation

1. Variante: "*LindEvol-GA*":

- 2-dim. Gitter
- keine zusätzlichen Zustandsparameter der Zellen
- keine samenproduzierenden Zellaktionen, Vermehrung von Pflanzen erfolgt allein durch den Selektionsmechanismus eines GA
- blockorientierte Genominterpretation
- keine bitweisen Mutationen, keine Duplikationen

eine Generation von Pflanzen wächst jeweils eine "Vegetationsperiode" lang gemeinsam in der Welt, diese besteht aus fester Zahl (**num_days**) von Wachstumsschritten ("Tagen").

- Startzustand jeder Vegetationsperiode: zufällig auf dem Boden verteilte Pflanzen, die nur aus 1 energielosen Zelle bestehen
- Fitnesswert einer Pflanze: Anzahl der energiereichen Zellen nach Ablauf der Vegetationsperiode



(*p* = Populationsgröße, *s* = Selektionsrate)

Typische Parameter:

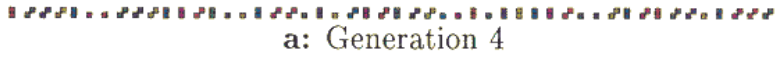
Populationsgröße: 50
Mutationsfaktor *m* = 2
Breite der Welt: 150
Höhe der Welt: 30
Genomlänge der Startpopulation: 20 Gene (40 Bytes)
Länge einer Vegetationsperiode: 30 Tage

Anzahl Generationen: 5000
Anzahl Aktionscodes für Wachstums-Aktionen: 224
Anzahl Aktionscodes für Mutationsratenveränderung: je 16

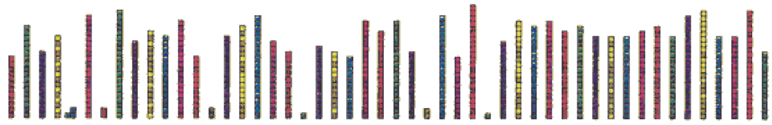
Mutationsrate: 0,01 (= "moderat")
Selektionsrate: 0,5 (= "moderat")

Ergebnis eines typischen LindEvol-GA-Laufs:

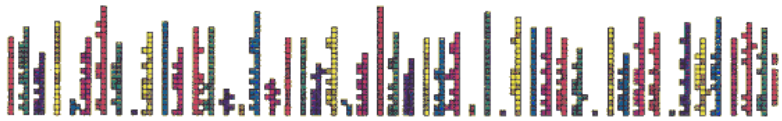
(jeweils Darstellung der gesamten "Welt" am Ende der Vegetations-periode, verschiedene Pflanzen durch verschiedene Farben markiert)



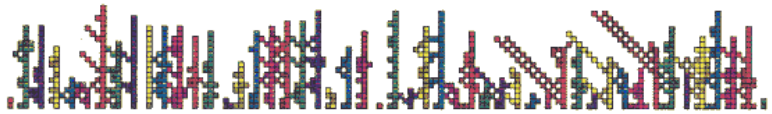
a: Generation 4



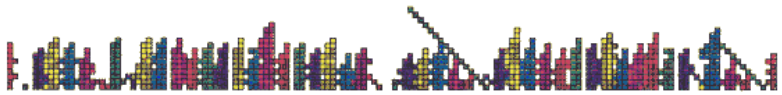
b: Generation 40



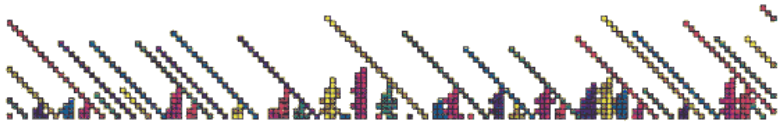
c: Generation 120



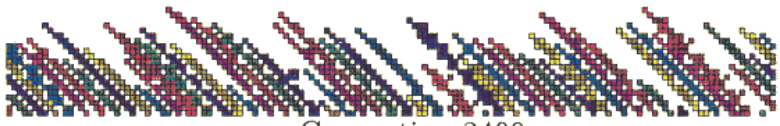
d: Generation 500



e: Generation 2200

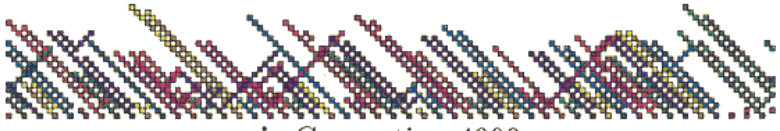


f: Generation 2207



g: Generation 2400

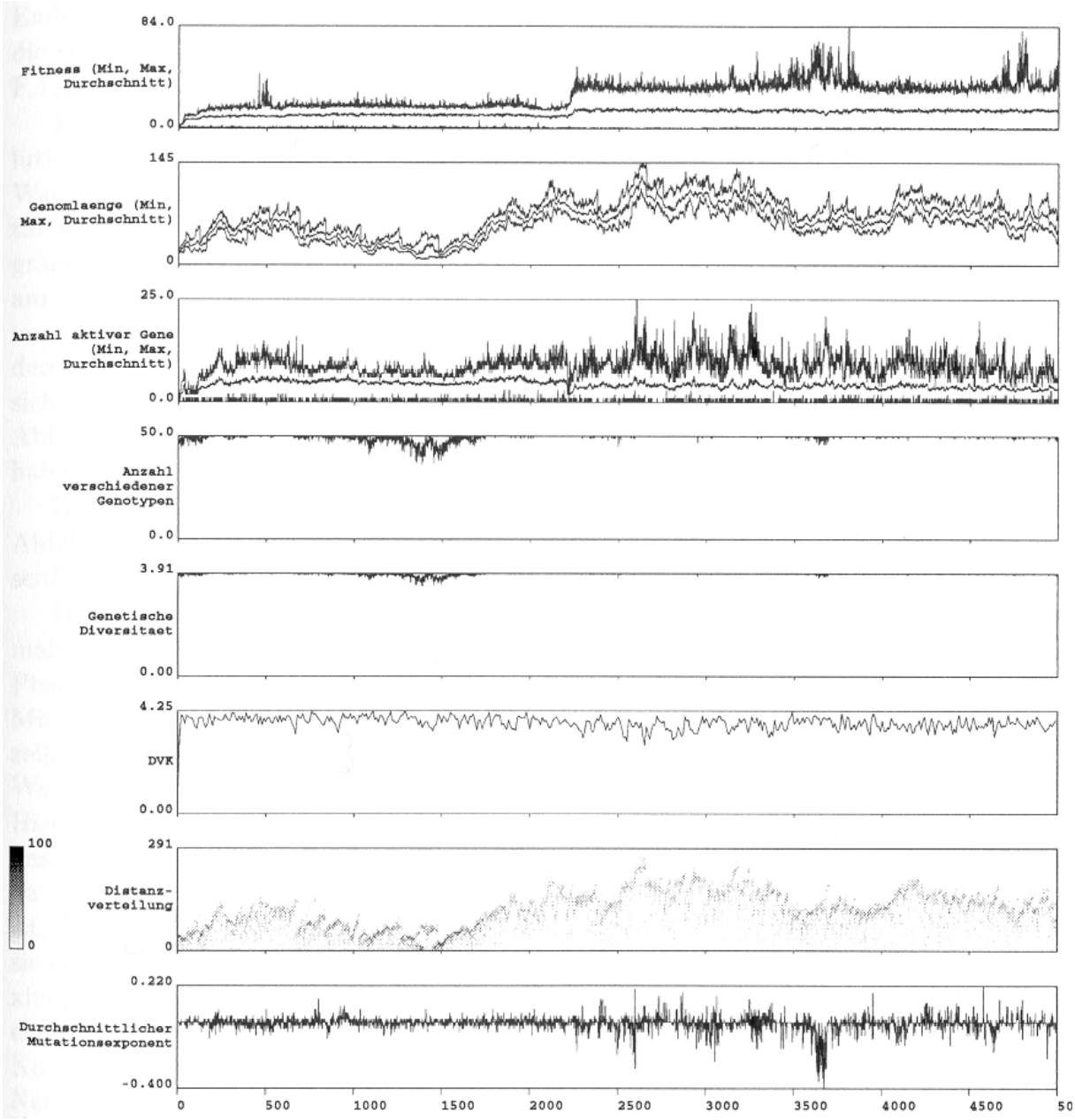
h: Generation 3608



i: Generation 4999

verschiedene "Phasen" der Evolution an Morphologien und Fitnesswerten erkennbar

statistische Übersicht des Simulationslaufs:



Anfangsphase (Generation < 30): kleine Pflanzen niedriger Fitness, genetisch festgelegte Größe

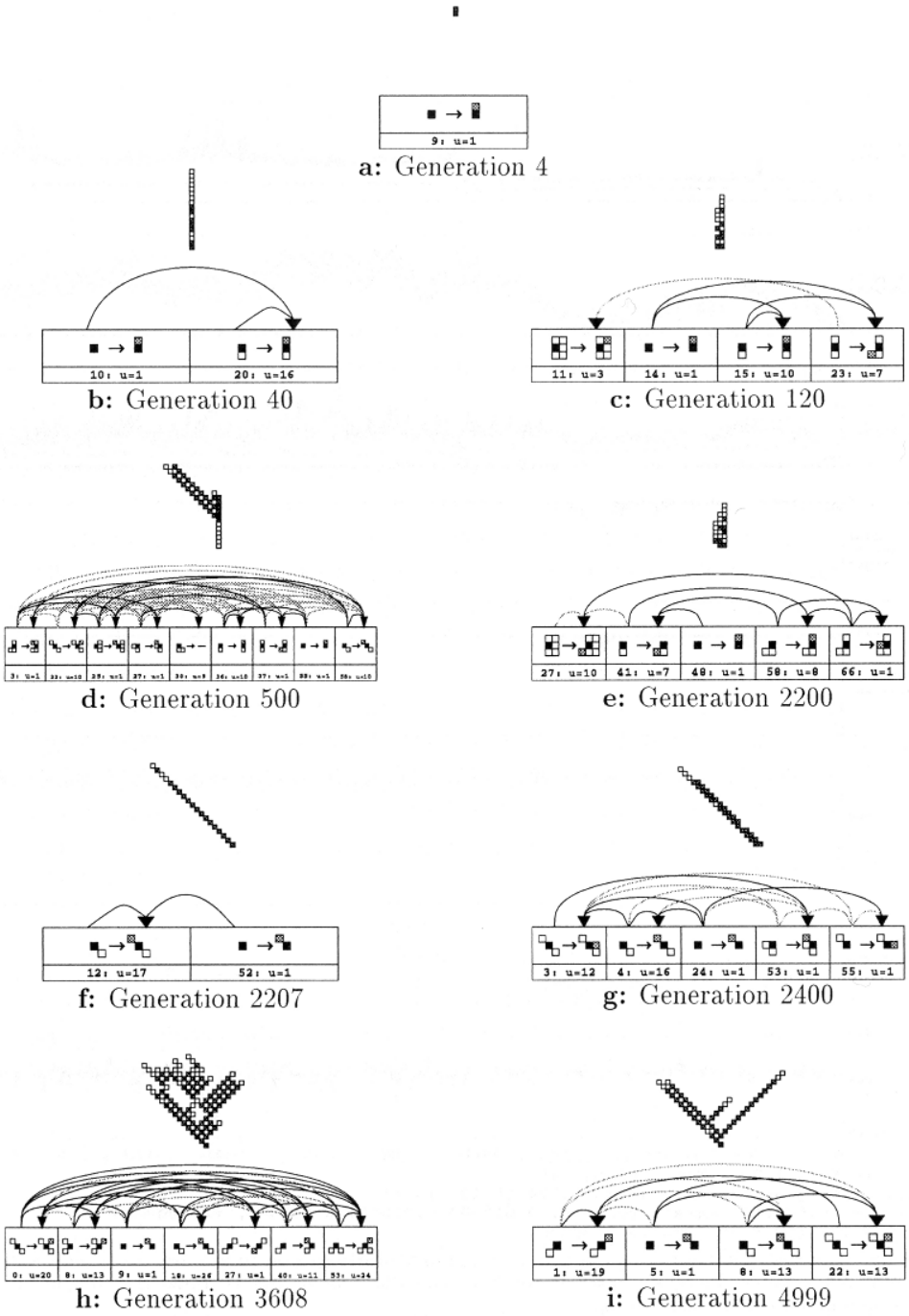
bei Gen. 30: Übergang zu unbeschränktem Wachstum ("evolutionärer Sprung")

Gen. 100: "Erfindung" von kleinen Seitenzweigen

Gen. 500: unbeschränkt wachsende diagonale Seitenzweige treten auf (aber: → räuml. Konkurrenz), verschwinden wieder

Gen. 2200: grundlegende Bauplanänderung: diagonaler Hauptspross, setzt sich durch (Anzahl der Zellen, die an einem Tag ein Photon absorbieren können, ist größer als bei vertikal wachsenden Pflanzen)
Gen. 2400: Verdickung des Hauptsprosses
Gen. 3500-3800: buschförmige Pflanzen mit vielen diagonalen Seitentrieben, verschwinden wieder

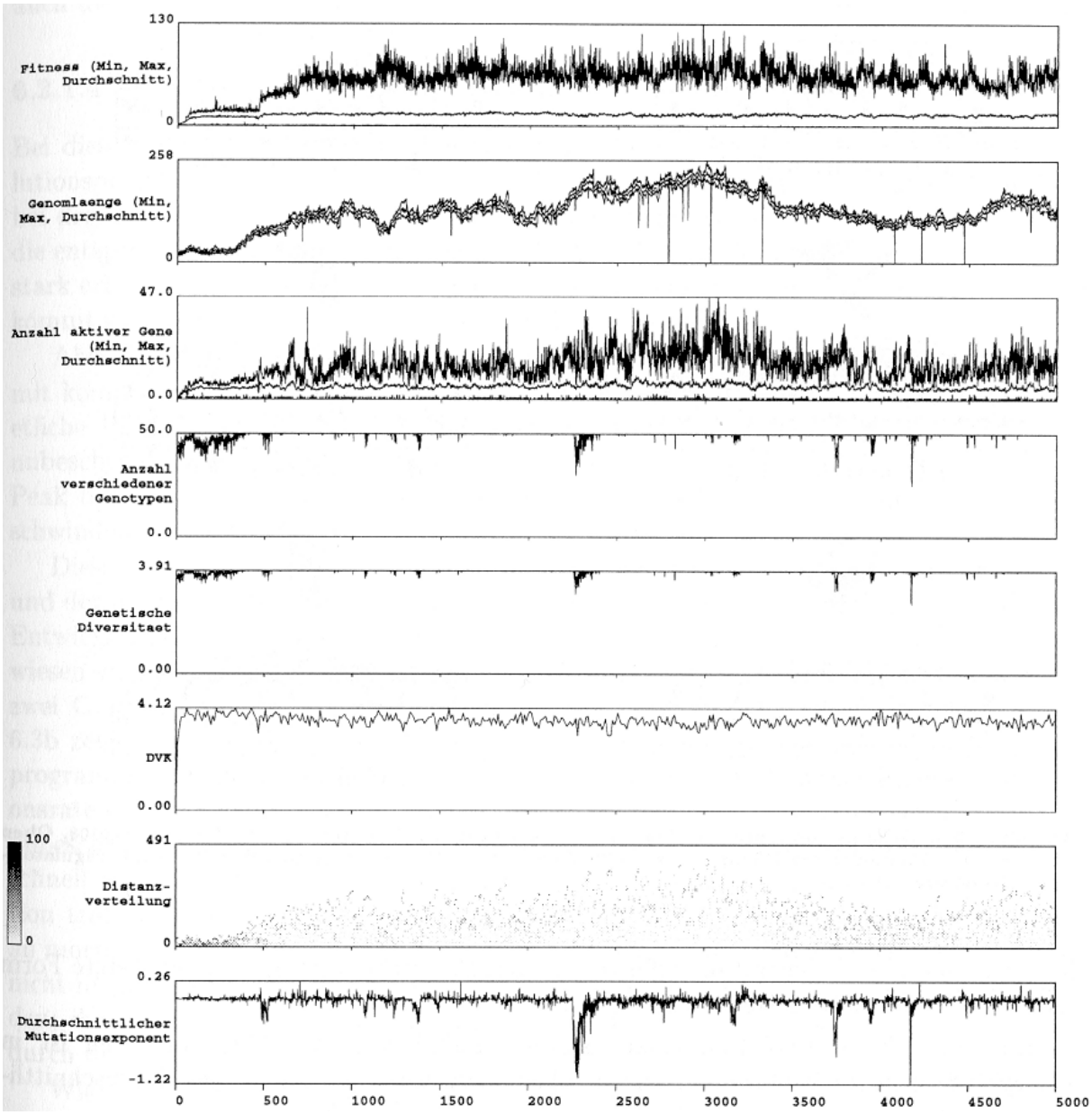
einige für die Phasen charakteristische Regelungsnetzwerke:



a: Generation 4
b: Generation 40
c: Generation 120
d: Generation 500
e: Generation 2200
f: Generation 2207
g: Generation 2400
h: Generation 3608
i: Generation 4999

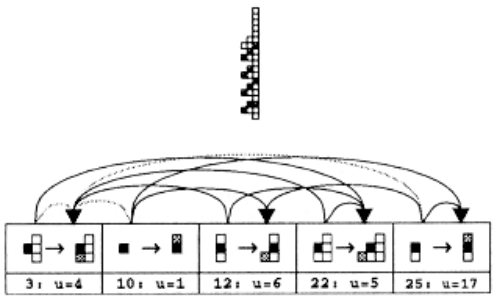
- es kommt zu einer Zunahme der Fitness und der morphologischen Komplexität
- breite Distanzverteilungen der Genome
- aber keine größere Distanzverteilungskomplexität als bei einem Kontrollversuch mit "neutraler Evolution" (= keine Heranziehung der Fitnesswerte bei der Selektion)
- Mutationsraten werden lediglich in einer kurzen Episode aktiv verändert

Vergleichslauf mit "scharfer Selektion" (Selektionsrate 0,8):

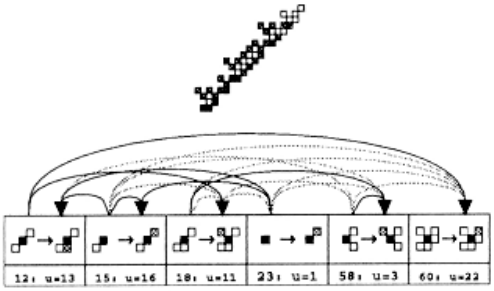


- ähnliche "Evolutionssprünge" wie im Lauf mit moderater Selektion
- Fitnesswerte erreichen höhere Maxima, aber keine höheren Durchschnittswerte
- buschartige Pflanzen erringen die Vorherrschaft
- bei scharfer Selektion notwendiger hoher Fitnesswert kann nur durch Überwuchern von Nachbarpflanzen erreicht werden
- höhere Genomlängen (erforderlich für die Busch-Architektur)

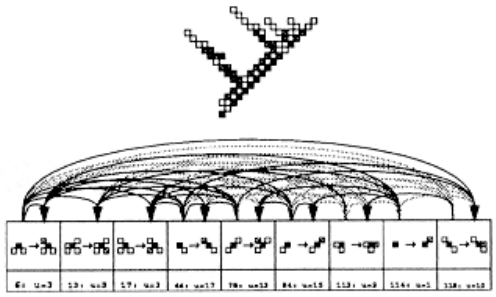
Beispiele für evolvierte Genome und Morphologien:

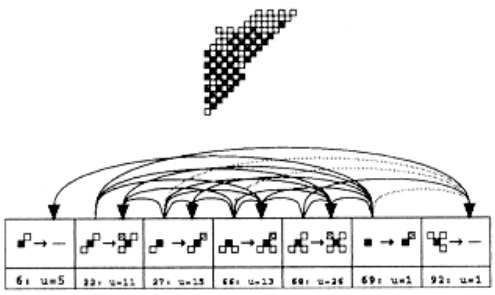


a: Generation 400

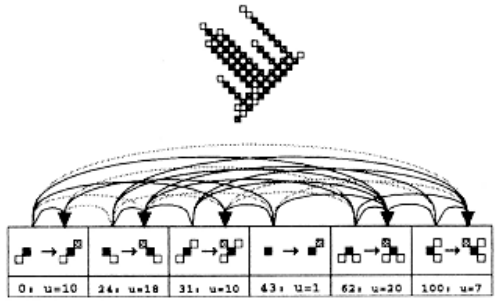


b: Generation 600

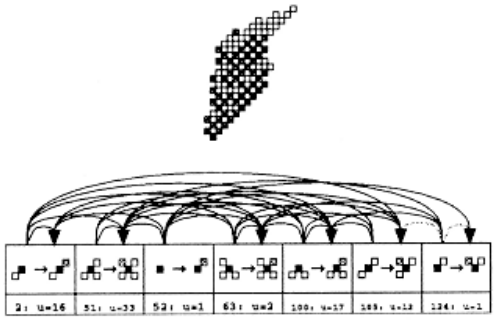

c: Generation 1000



d: Generation 2000



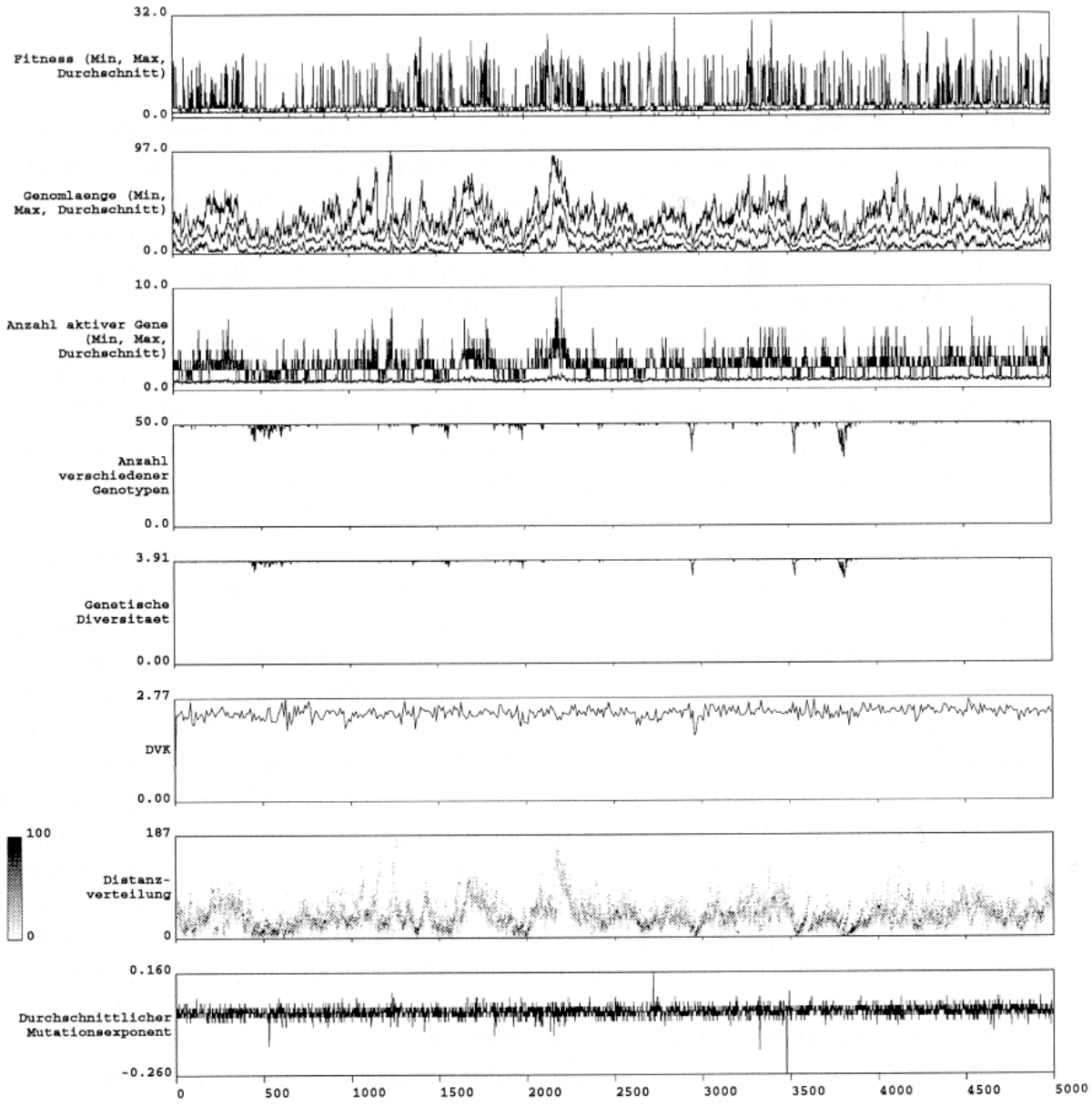
e: Generation 4000



f: Generation 4999

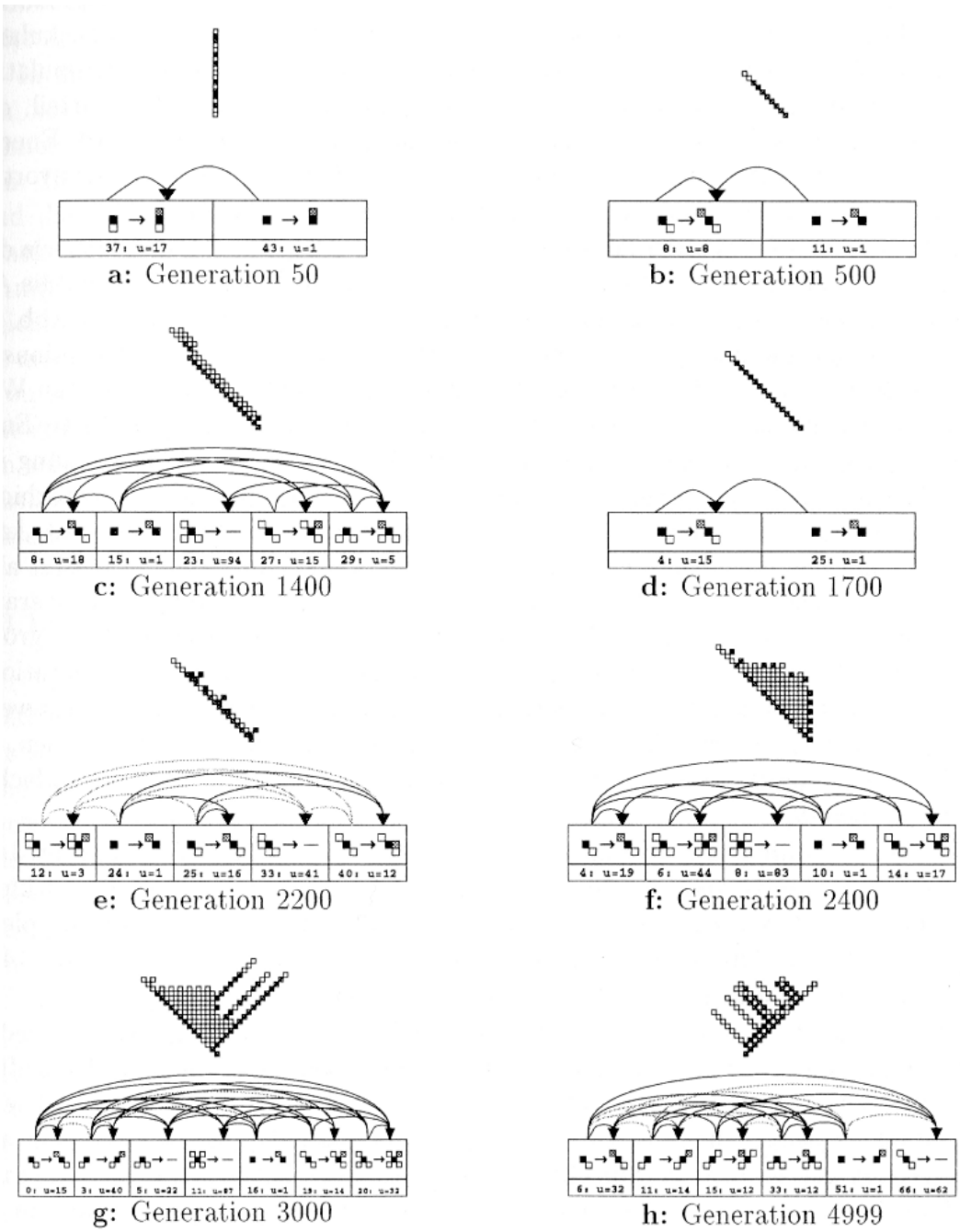
Vergleichslauf mit hoher Mutationsrate (0,1) (und moderater Selektion):

- keinerlei Entwicklung von Formen mit komplexerer Morphologie
- auch nach 5000 Generationen existieren etliche Pflanzen, die überhaupt nicht wachsen
- unbeschränkt wachsende Formen treten gelegentlich auf, aber verschwinden wieder trotz Selektionsvorteil
- "kritische Fehlergrenze" der Mutation überschritten
- große Distanzen zwischen den Genomen, kleine DVK

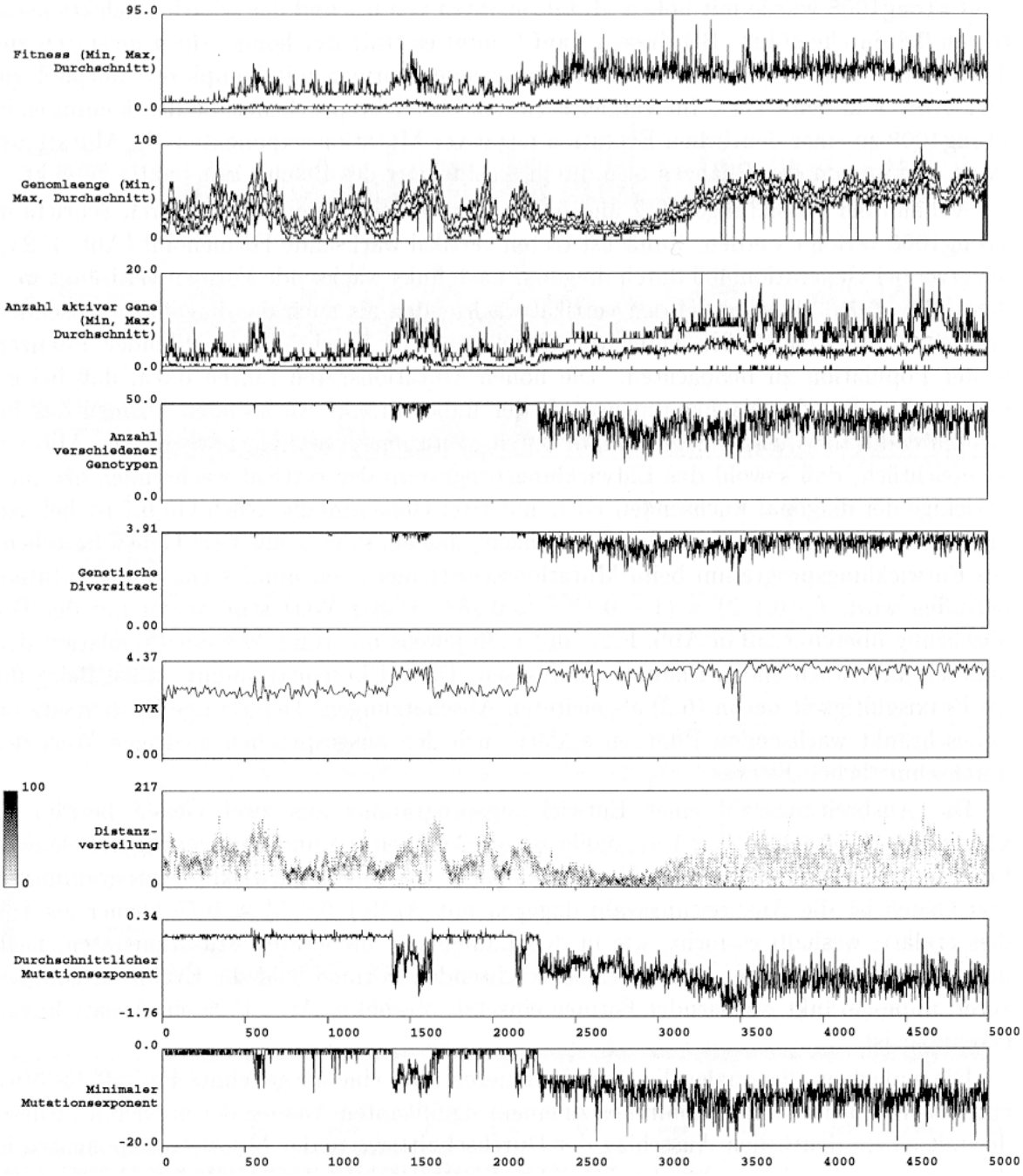


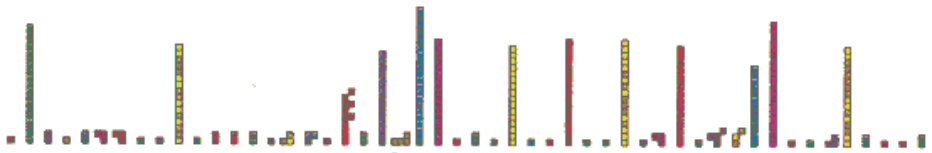
Kombination von hoher Mutationsrate (0,1) und scharfer Selektion (0,8):

- dauerhafte Etablierung unbeschränkt wachsender Formen mit komplexer Morphologie
- deutliche Evolution negativer Mutationsexponenten (aktive Senkung der Mutationsraten)!
- ca. bei Generation 1400 entstehen Pflanzen mit zweiter Zellschicht, "Knicken" in der Achse und Genen mit "Mehrfachfunktion" (s. Abb. unten, c); Zellen der aufgelagerten Schicht führen Aktion "**`mut-`**" durch

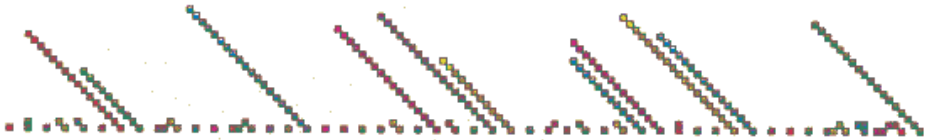


- mutationsratensenkende Form hat keinen unmittelbaren Selektionsvorteil durch höhere Fitness, sondern aufgrund größerer Ausbreitungsrate ihrer Genome durch die niedrigeren Mutationsraten
- Dominanz mutationsratensenkender Formen geht bei Generation 1650 aus ungeklärten Gründen zuende
- bei Generation 2200 treten wieder solche Formen auf, entwickeln sich weiter zu buschigen Formen
- deutlicher Anstieg der DVK, wenn Mutationsraten aktiv gesenkt werden

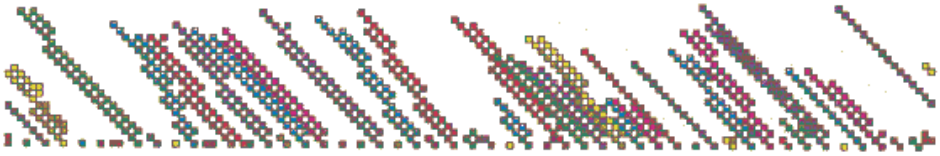




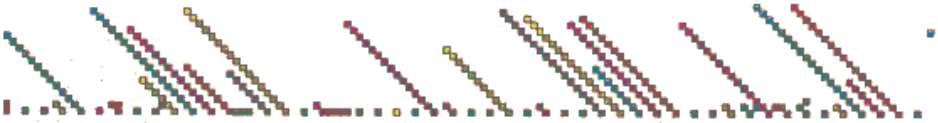
a: Generation 50



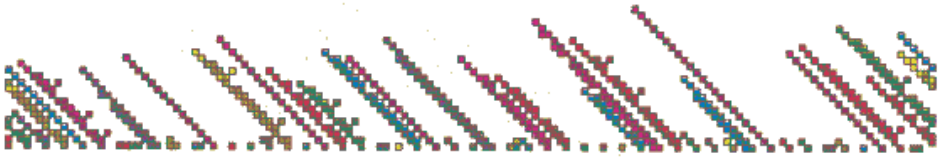
b: Generation 500



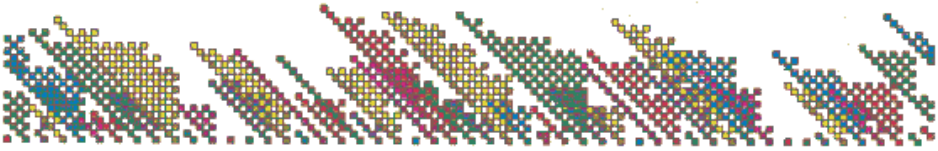
c: Generation 1400



d: Generation 1700

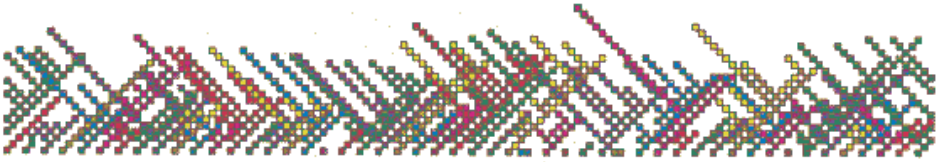


e: Generation 2200



f: Generation 2400

g: Generation 3000



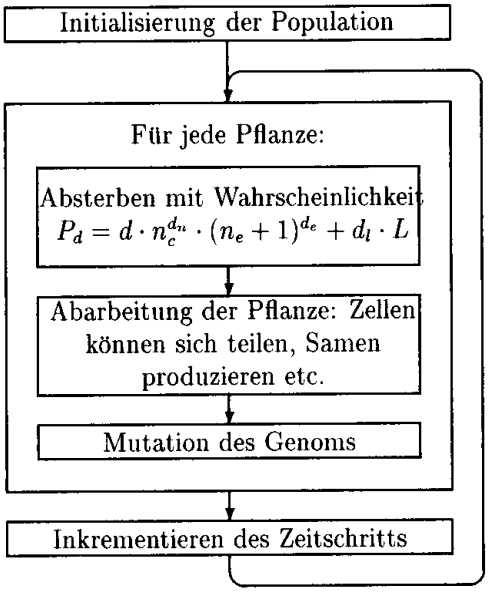
h: Generation 4999

- Evolution von Genomen mit aktiv gesenkten Mutationsraten heißt: Evolution zur "edge of chaos" (kritische Fehlergrenze), von der "Chaos"-Seite her!

2. Variante: "*LindEvol-B*"

- 2-dim. Gitter; nur Energieparameter, kein interner Zustand (wie in LindEvol-GA)
- blockorientierte Genominterpretation; bitweise Änderungsmutationen, Insertionen, Deletionen
- zusätzliche Zellaktionen: Samenausbreitung (lokal und fernwirkend)
- kein starrer Selektionsmechanismus in Abhängigkeit von Energiegehalt; Vermehrung erfolgt durch Samenausbreitung
- Pflanzen sterben durch zufallsabhängigen Tod oder durch Angriff; keine Synchronisierung der Lebensphasen!
- Wahrscheinlichkeit für zufallsabhängigen Tod hängt ab von Größe, "Überhangskoeffizient" (räuml. Asymmetrie) und Energiegehalt der Pflanze
- "Angriff" erfolgt, wenn bei Besetzung einer Nachbarzelle diese bereits besetzt ist
- Wahrscheinlichkeit für Erfolg eines Angriffs ist umgekehrt proportional zur Gesamtenergie der angegriffenen Pflanze – im Erfolgsfall stirbt die angegriffene Pflanze

- keine doppelte Zeit-Schleife mehr:



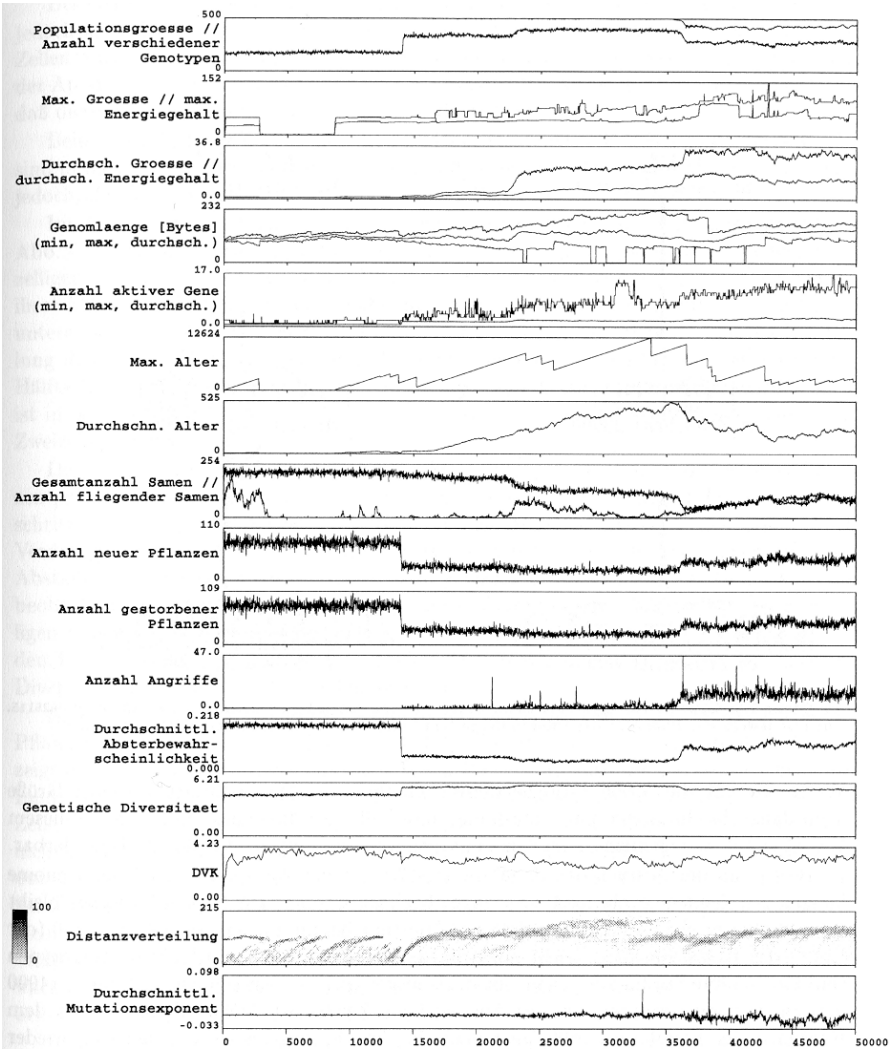
Konsequenzen:

- die Populationsgröße ist veränderlich
- die Pflanzen können auch alle aussterben
- eine einzelne Pflanze kann die ganze Welt überwuchern
  – solche "superstabilen" Pflanzen entstehen tatsächlich:

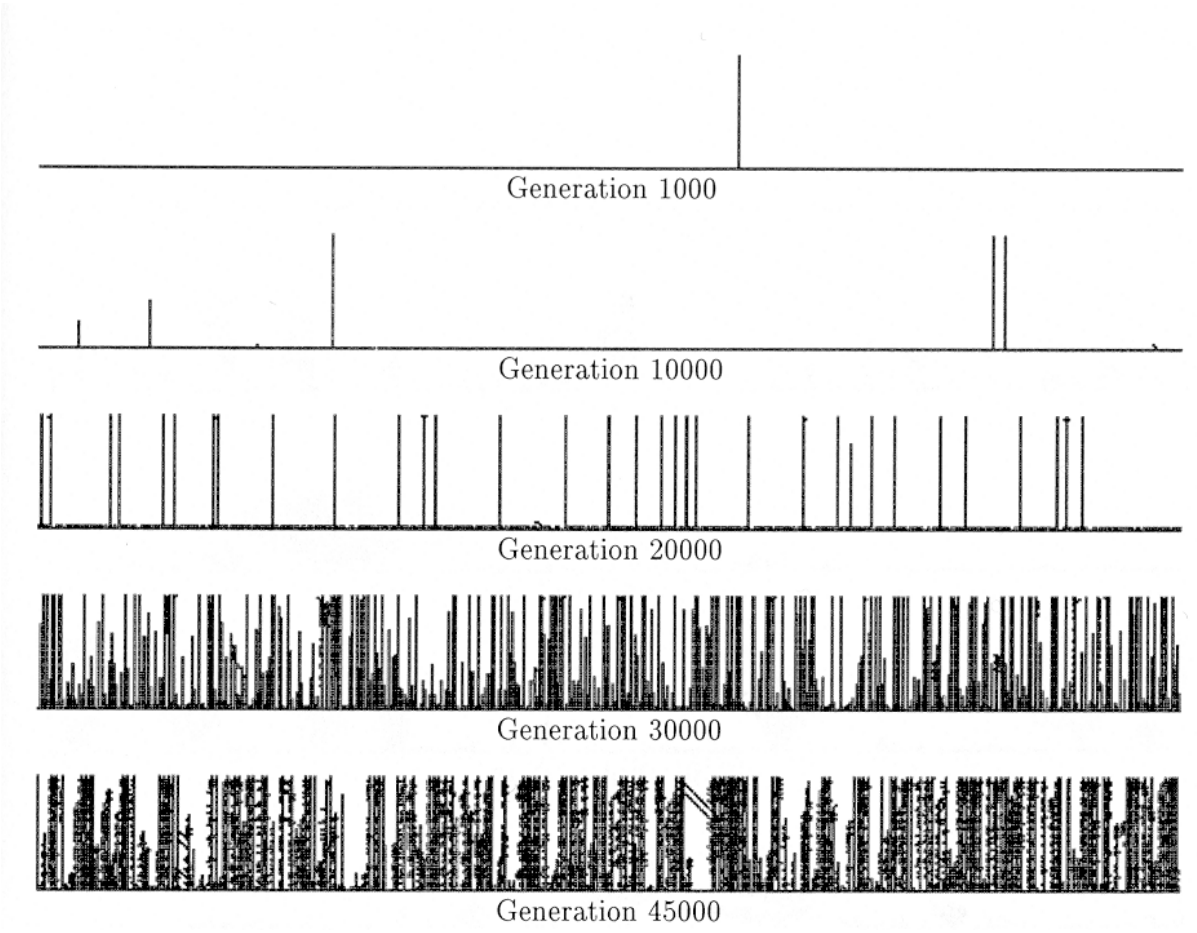
a: $d_l = 0.02$

- bei kleinen Werten der Absterbewahrscheinlichkeit sind primitive, einzellige Formen sehr stabil; bei zu hohen Werten stirbt dagegen die gesamte Population in der Anfangsphase
- Startpopulation darf nicht zu klein sein (gewählt wurde N = 500), da ein großer Anteil der initialen Pflanzen nicht überlebensfähig ist

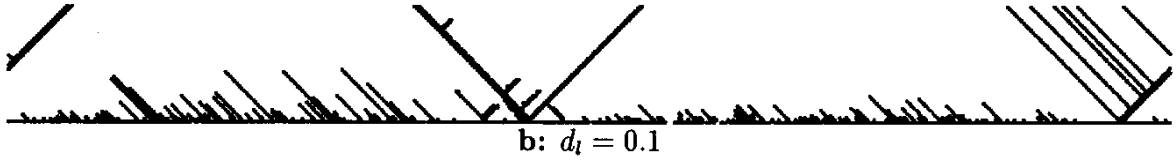
Beispiel-Lauf:



- verschiedene Evolutionssprünge
- am Anfang sind die vertikalen Pflanzen "steril", erreichen aber sehr hohes Alter
- erst nach 25000 Zeitschritten treten unbeschränkt wachsende, vermehrungsfähige Formen auf
- Raumkonflikte und Angriffe treten erst nach Zeitschritt 35000 im nennenswerten Ausmaß auf



bei anderer Parametrisierung können auch diagonal wachsende Formen die Vorherrschaft erringen:

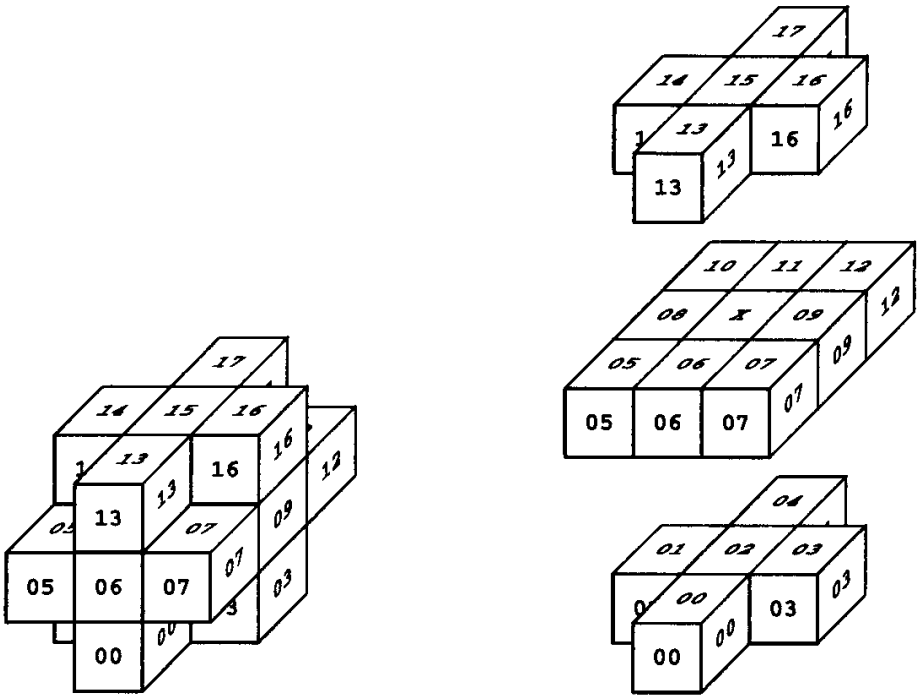


b: $d_l = 0.1$

(nach ca. 50000 Schritten)

- aktive Änderung der Mutationsraten trat in geringerem Umfang auf als in LindEvol-GA

Versuch der Verallgemeinerung auf 3 Dimensionen:
18-Zellen-Nachbarschaft



Übertragung der blockorientierten Genominterpretation (nun mit 18-Bit-Codierung der lokalen Situation einer Zelle):
scheitert!

keine Evolutionsprozesse; die Anfangspopulation stirbt stets aus, ohne Nachkommen zu produzieren

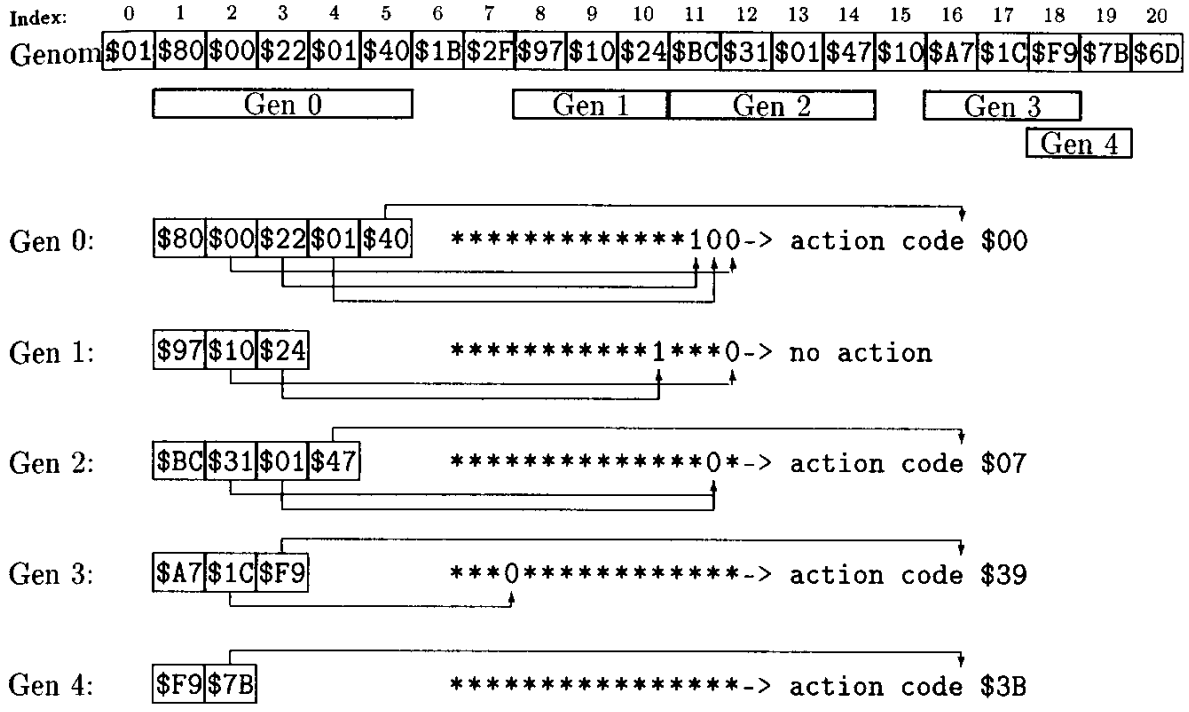
- Wahrscheinlichkeit für Aktivierung eines Gens ist bei 18 Bit-Zellzuständen zu gering
- Generierung eines "Keimzellgens" zur Erzeugung einer vermehrungsfähigen Pflanze ist zu unwahrscheinlich
- 1000-fach größere Startpopulation wäre nötig (zu rechenintensiv)

Ausweg: "promotor-orientierte Genominterpretation"

- analog zur Natur: Markierung von Anfang und Ende eines Gens durch Promotor- und Terminator-Regionen (mit speziellen Codes)
- Gene haben variable Länge
- Gene können sich gegenseitig überlappen

Realisierung der promotor-orientierten Genominterpretation in LindEvol:

- Promotor = Byte, in dem Bit 7 gesetzt ist
- Terminator = Byte, in dem Bit 6 gesetzt ist
- terminiertes Gen = Abschnitt von einem Promotor bis zu einem Terminator, in dem sich keine Promotoren oder Terminatoren befinden
- nichtterminiertes Gen = Abschnitt zwischen 2 Promotoren (oder von einem Promotor bis zum Ende des Genoms), in dem keine Promotoren oder Terminatoren liegen
- bei terminierten Genen steht in den Bits 0-5 des Terminators der Aktionscode
- nichtterminierte Gene lösen keine Aktion aus
- linke Regelseiten können jetzt auch Variablen (wildcard-Symbol *) enthalten
- jedes Byte zwischen einem Promotor und einem Operator (das nicht selbst Promotor oder Terminator ist) spezifiziert ein Bit der linken Regelseite



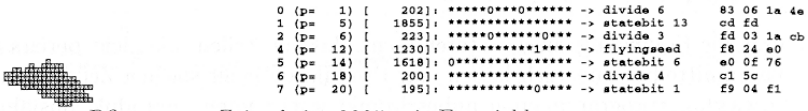
3. Variante: "*LindEvol-P*"

- 2-dim. Gitter
- Zellen mit internem 16-Bit-Zustandsparameter
- Zellaktionen: wie in LindEvol-B, zusätzlich "Zustand setzen"
- promotororientierte Genominterpretation
- Mutationen: bitweise Änderungen, byteweise Insertionen und Deletionen, Genduplikationen
- Vermehrung, Absterben, Angriff wie in LindEvol-B

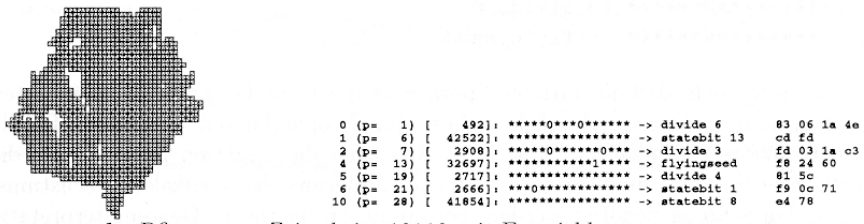
Ergebnisse:

- in vielen Fällen entwickelt sich wesentlich komplexere Dynamik als in den bisherigen Varianten
- flexible Längen und Genspezifikation mit wildcards führt zu deutlich besserer Evolvierbarkeit
- aber größerer Rechenaufwand

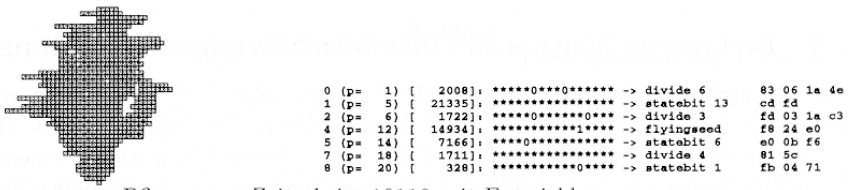
Beispiel-Pflanzen aus einem Simulationslauf:



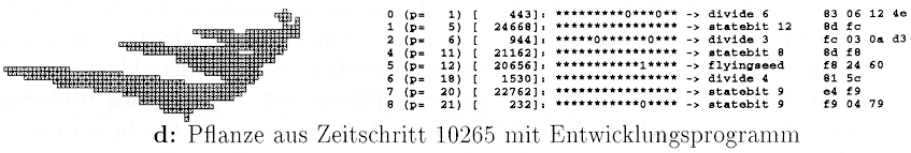
a: Pflanze aus Zeitschritt 9995 mit Entwicklungsprogramm



b: Pflanze aus Zeitschritt 10110 mit Entwicklungsprogramm

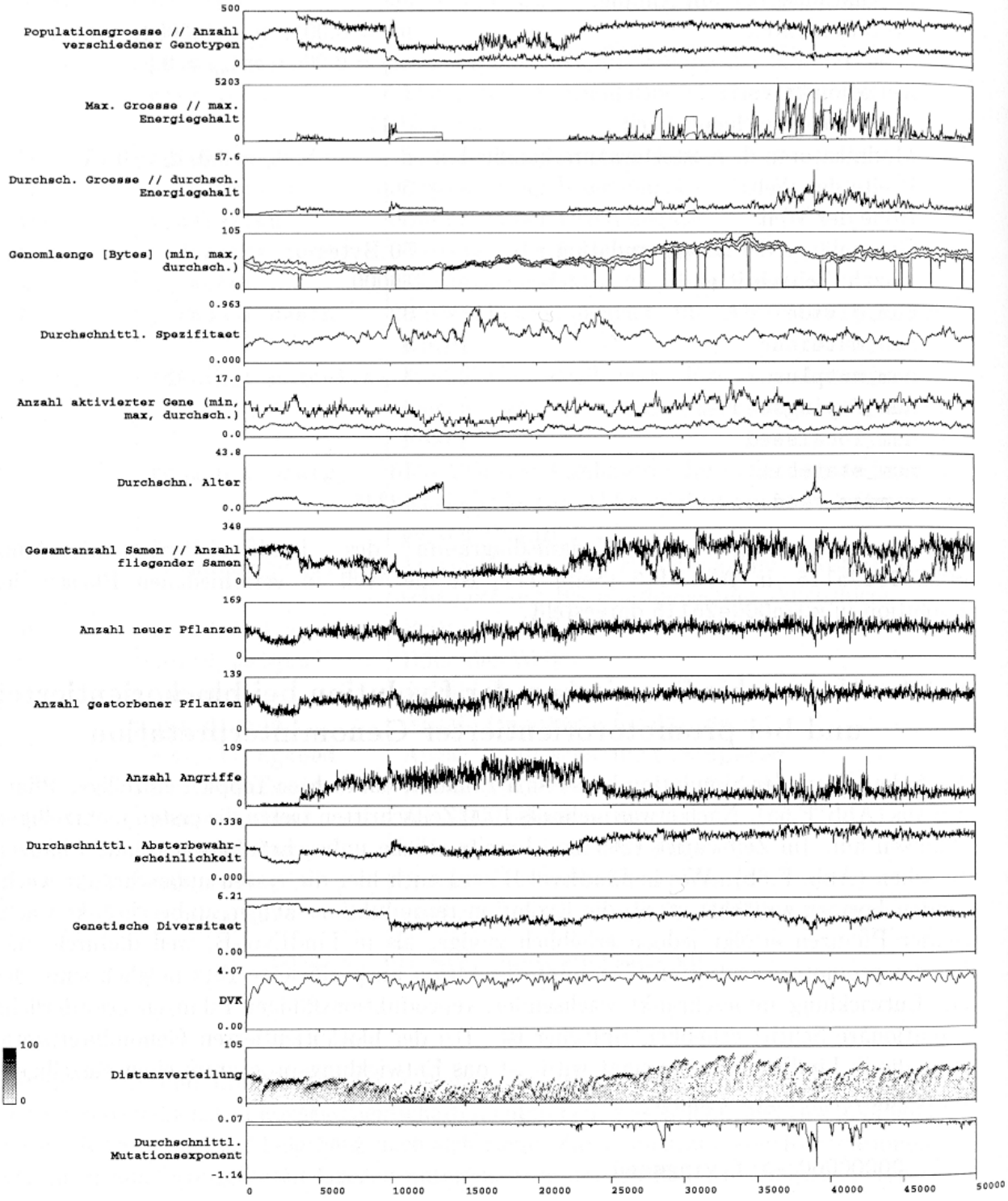


c: Pflanze aus Zeitschritt 10110 mit Entwicklungsprogramm



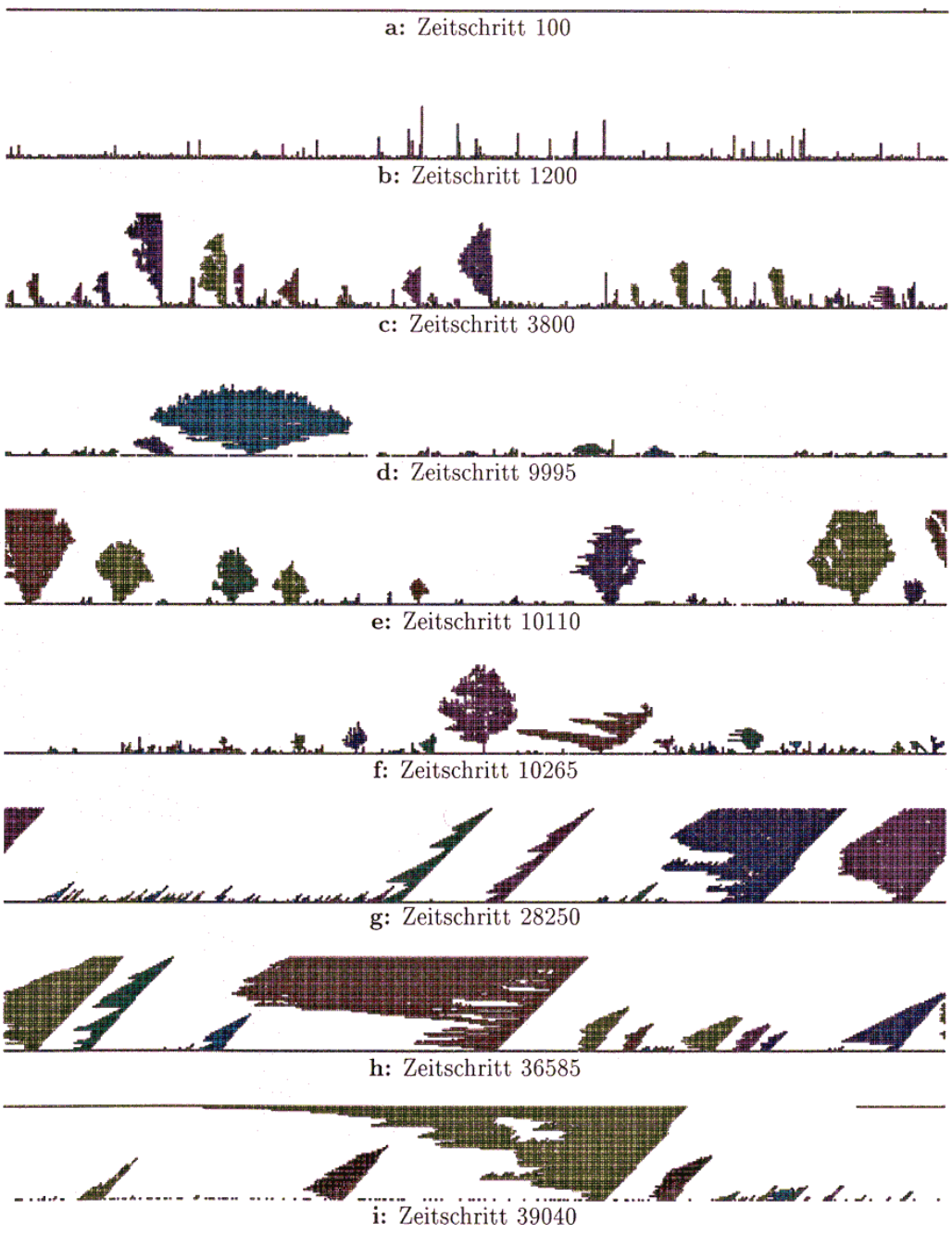
d: Pflanze aus Zeitschritt 10265 mit Entwicklungsprogramm

Verlaufsdiagramme:



- auf die (relativ schnelle) Evolution unbeschränkt wachsender Pflanzen folgt als weiterer Schritt das Auftreten von Pflanzen mit Breitenausdehnung (zunächst nur nach links)
- beidseitig sich ausbreitende Strukturen kurz vor Zeitschritt 10000

- busch- und baumartige Pflanzen sterben vor Schritt 11000 aus, werden von "angriffslustigen" Formen verdrängt, die nur horizontal am Boden entlangwachsen
- bei Schritt 22000 diagonal nach rechts wachsende Formen
- ab Schritt 36000 Riesenformen, die z.T. Energiekrisen auslösen
- ab Schritt 45000 wieder weniger Riesenpflanzen und weniger Angriffe (→ Evolution kooperativeren Wachstumsverhaltens?)



a: Zeitschritt 100

b: Zeitschritt 1200

c: Zeitschritt 3800

d: Zeitschritt 9995

e: Zeitschritt 10110

f: Zeitschritt 10265

g: Zeitschritt 28250

h: Zeitschritt 36585

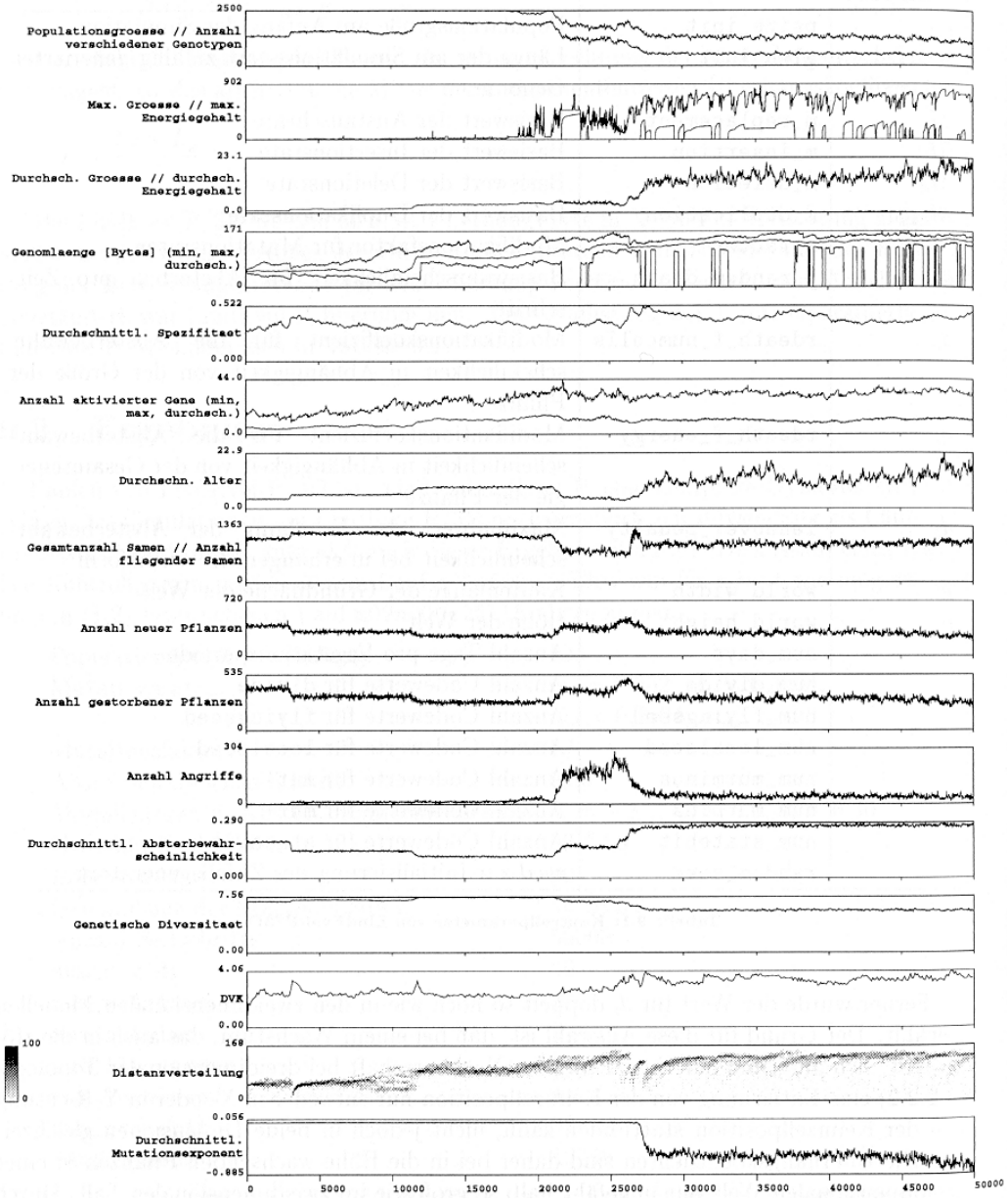
i: Zeitschritt 39040

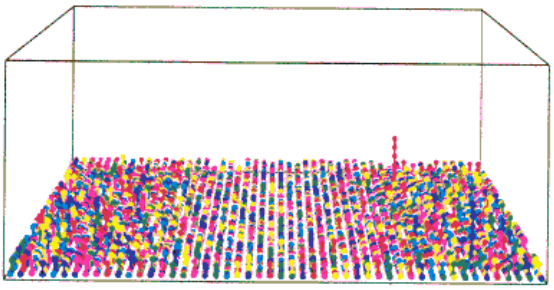
4.Variante: "LindEvol-P/3D" (3-dim. Version von LindEvol-P)

- 3-dim. Gitter
- interner Zellzustandsparameter mit 32 Bit
- Überhangskoeffizient als Mittelwert aus komponentenweisen Überhangskoeffizienten (für Absterbewahrscheinlichkeit)
- sonst wie dei 2D-Version

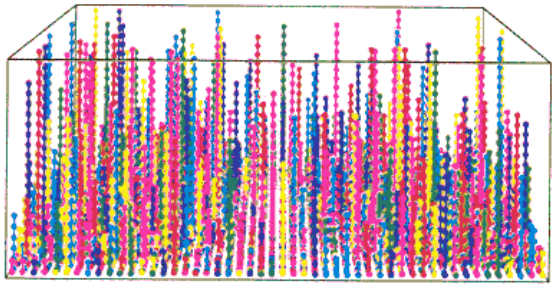
Ergebnisse:

- komplexe Dynamik; Auftreten der "fortgeschrittenen" Formen erfolgt aber langsamer als in 2D
- aktives Absenken des Mutationsexponenten ab ca. Schritt 26000

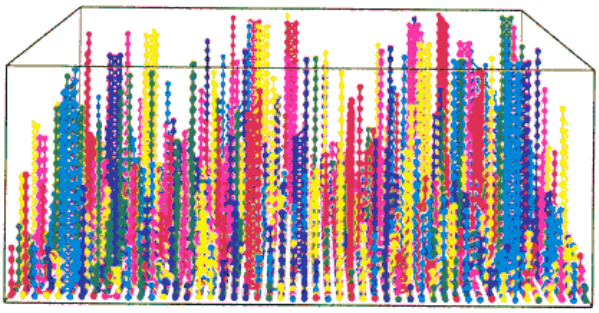


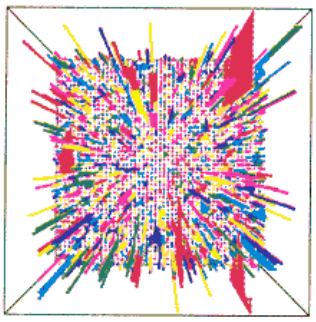


a: Zeitschritt 8000

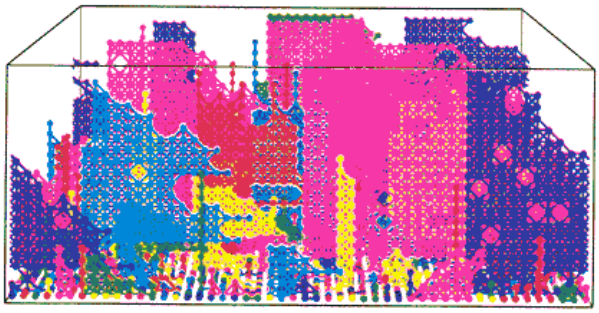


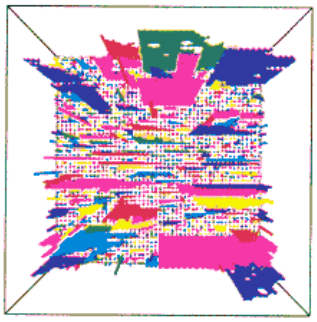
b: Zeitschritt 16000



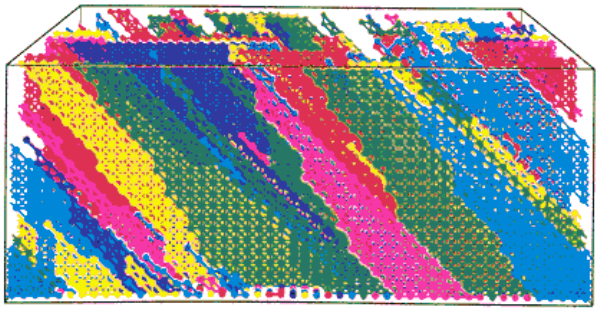


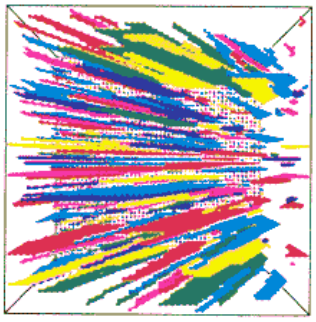
c: Zeitschritt 20000, Vorderansicht und Vogelperspektive





d: Zeitschritt 24000, Vorderansicht und Vogelperspektive





e: Zeitschritt 40000, Vorderansicht und Vogelperspektive