Βελτιστοποίηση

Ιωάννης Γ. Τσούλος

Τμήμα Πληροφορικής και τηλεπικοινωνιών
Πανεπιστήμιο Ιωαννίνων

2022

Περίληψη

Θεωρία της εξέλιξης των ειδών (1)

1. Διατυπώθηκε τον περασμένο αιώνα από τον Δαρβίνο.
2. Πολεμήθηκε από την Εκκλησία και τους συντηρητικούς εκείνης της εποχής
3. Ακόμα και σήμερα υπάρχουν άνθρωποι που δεν την δέχονται

Θεωρία της εξέλιξης των ειδών (2)

1. Δεν υπάρχει διαχωρισμός σε ανώτερα και κατώτερα είδη
2. Το περιβάλλον καθορίζει το πλήθος των απογώνων
3. Τα χαρακτηριστικά των ειδών αποθηκεύονται σε χρωμοσώματα
4. Τα χρωμοσώματα αποτελούνται από γονίδια
5. Οι βασικές λειτουργίες των οργανισμών είναι η **αναπαραγωγή** και η **μετάλλαξη**.

Γενότυπος - φαινότυπος

1. Το σύνολο της πληροφορίας που αποθηκεύεται στα γονίδια ονομάζεται **γενότυπος**.
2. Τα χαρακτηριστικά ενός οργανισμού σχηματίζονται με την αποκωδικοποίηση του γενότυπου.
3. Τα χαρακτηριστικά που είναι ορατά μετά από την αποκωδικοποίηση είναι ο φαινότυπος.

Παράδειγμα Γενότυπου - φαινότυπου

1. DNA: μια σειρά από γράμμματα.
2. Χρώμα ματιών: Η αποκωδικοποίηση ενός μικρού μέρους του DNA

Ιστορική αναδρομή

1. Ξεκίνησαν το 1975 από τον Holland
2. Στηρίχτηκαν στην θεωρία εξέλιξης των ειδών
3. Ομάδα από υποψήφιες λύσεις:
4. Εξέλιξη των λύσεων με διασταύρωση, μετάλλαξη μέχρι να υπάρξει σύγκλιση
5. Αρχικά εφαρμόστηκαν σε δυαδικά προβλήματα
6. Διατυπώθηκαν και θεωρήματα σύγκλισης.

Αντιστοιχίσεις

1. Γονίδιο => Στοιχείο πίνακα
2. Χρωμόσωμα => Πίνακας
3. Γενότυπος => Πίνακας (παράμετρος x)
4. Φαινότυπος => συνάρτηση $f(x)$ => Καταλληλότητα
5. Διασταύρωση => Ανταλλαγή στοιχείων μεταξύ πινάκων
6. Μετάλλαξη => Τυχαία αλλαγή στοιχείων σε πίνακες

Βασικό σχήμα γενετικών αλγορίθμων

```
Procedure Genetic
   t=0
   Initialize ( P(t) )
   Evaluate ( P(t) )
   while (not terminated (P(t)) )do
     t=t+1
     subP(t)=Select( P(t) )
     Crossover( subP(t) )
     Mutate ( subP(t) )
     P(t+1) = Recombine (P(t), subP(t))
   end while
End Genetic
```

Παρατηρήσεις

1. t είναι ο αριθμός επανάληψης ή και γενία.
2. P(t) είναι ο πληθυσμός των λύσεων στην γενιά t
3. Δεν είναι απαραίτητο να γίνουν όλα τα βήματα
4. Δεν είναι απαραίτητο να είναι αυτή η σειρά των βημάτων
5. Σημαντικό ρόλο παίζει το κριτήριο τερματισμού

Υβριδικοί γενετικοί

1. Λαμαρκιανοί
   1. Μετά το τέλος του αλγορίθμου εφαρμόζεται διαδικασία τοπικής βελτιστοποίησης
   2. Γρήγορη μέθοδος
   3. Πιο συχνοί
2. Δαρβινικοί
   1. Εφαρμόζεται η τοπική μέθοδος μετά από κάθε υπολογισμό καταλληλότητας ή περιοδικά
   2. Αργή μέθοδος
   3. Καλύτερα αποτελέσματα

Πλεονεκτήματα

1. Γρήγοροι και αξιόπιστοι
2. Συνεργάζονται εύκολα με άλλα συστήματα
3. Επεκτείνονται εύκολα
4. Εφαρμόζονται σχεδόν σε κάθε πεδίο επιστήμης
5. Είναι ανθεκτικοί σε σφάλματα
6. Μπορούν να παραλληλοποιηθουν εύκολα

Βασικές αρχές

1. Η αναπαράσταση είναι ο θεμέλιος λίθος των γενετικών.
2. Θα πρέπει να μην χάνετα πληροφορία κατά την αναπαράσταση δεδομένων (πχ από μετατροπή δεκαδικών τιμών με υποδιαστολή σε δυαδική αναπαράσταση).
3. Θα πρέπει να είναι φανερή η διάκριση ανάμεσα σε καλά και κακά χρωμοσώματα.

Δυαδική αναπαράσταση

1. Είναι η πρώτη αναπαράσταση που χρησιμοποιήθηκε.
2. Εμφανίζεται σε δύο μορφές:
   1. Όλο το χρωμόσωμα είναι ένας ακέραιος αριθμός σε δυαδική μορφή
   2. Το χρωμόσωμα αποτελείται από πολλούς δυαδικούς αριθμούς (πχ παραμέτρους συνάρτησης)

Παράδειγμα δυαδικής αναπαράστασης

1. **Πρόβλημα**: Να βρεθεί εκείνος ο ακέραιος στο διάστημα [0,255] με την μέγιστη μετάβαση από 0 σε 1 στα bits.
2. **Χρήση δυαδικής αναπαράστασης**: πχ ο αριθμός 45 αναπαρίσταται σαν $101101_{(2)}$.
3. Στον αριθμό 45 η καταλλήλοτητα είναι 2 (έχουμε δύο μεταβάσεις).
4. Ο αριθμός μπορεί να αναπαρασταθεί και με 8 bits: $00101101_{(2)}$.
5. Η λύση στο ζητούμενο πρόβλημα είναι ο αριθμός 85 ή $01010101_{(2)}$ με 4 μεταβάσεις.

Μετατροπή ακεραίων στο δυαδικό σύστημα

1. Κάθε αριθμός στο δυαδικό σύστημα αναπαρίσταται σαν $X = \sum_{i=0}^{N} b_i 2^i$
2. Το N καθορίζεται από την αρχιτεκτονική των υπολογιστών πχ. 32bit ή 64bit.
3. Τα στοιχεία b_i μπορούν να έχουν τιμές 0 ή 1 μόνον.
4. Τα ψηφία γράφονται από την μεγαλύτερη δύναμη προς την μικρότερη: πχ $45_{(10)} = 101101_{(2)} = 2^5 + 2^3 + 2^2 + 2^1 = 45$

Αλγόριθμος μετατροπής στο δυαδικό σύστημα

```
Procedure convert(x)
  i=1
  while (x>0) do
    y(i) = x mod 2
    x = x/2
    i = i+1
  end while
  reverse(y)
End convert
```

Πρόβλημα δυαδικής αναπαράστασης

1. Μικρές αλλαγές προκαλούν μεγάλες αλλαγές στην καταλληλότητα.
2. **Παράδειγμα**: Ο αριθμός $15_{(10)} = 01111_{(2)}$ και ο $16_{(10)} = 10000_{(2)}$. Απέχουν 1 αριθμό αλλά σε δυαδική αναπαράσταση απέχουν 5.
3. **Λύση** σε τέτοια προβλήματα: Η αναπαράσταση Gray.

Κωδικοποίηση Gray

1. Δυαδική αναπαράσταση αριθμών.
2. Διαδοχικοί αριθμοί διαφέρουν κατά 1 bit.
3. Χρησιμοποιούνται αρκετά στην Ασφάλεια Πληροφοριών.
4. Υπάρχουν πολλές κωδικοποιήσεις Gray, εξαρτάται πως ξεκινάμε κάθε φορά.

Παράδειγμα κωδικοποίησης Gray

Decimal	Binary	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0101
6	0110	0101
7	0111	0100
8	1000	1100

Μετατροπή δυαδικού σε Gray

```
Procedure bin2gray(g,b)
  g(1)=b(1)
  for k=2 to numberofbits(g) do
    g(k) = b(k-1) XOR b(k)
  end for
end bin2gray
```

Μετατροπή Gray σε δυαδικό

```
Procedure gray2bin(b,g)
 value=g(1)
 b(1) = value
 for k=2 to numberofbits(g) do
   if(g(k)=1) then
     value=NOT value
   end if
   b(k)=value
 end for
end gray2bin
```

Περιπτώσεις αναπαράστασης ακεραίων

1. Έστω ότι διαθέτουμε N δυαδικά στοιχεία (θέσεις σε πίνακα).
2. Αν $x \in \left[0 \ldots 2^{N-1}\right]$ ο αριθμός αναπαρίσταται άμεσα με δυαδική μετατροπή.
3. Αν $x \in \left\{M \ldots M + 2^N - 1\right\}$ ο αριθμός αναπαρίσταται άμεσα αλλά θα πρέπει να γίνεται ολίσθηση κατά M όταν θα υπολογίζεται η καταλληλότητα. Δηλαδή, ο αριθμός μπορεί να αναπαρασταθεί στο δυαδικό σύστημα, αλλά οι θέσεις [0..M-1] δεν αναπαριστουν κάτι. Παράδειγμα έστω ότι ένα θερμότρο σε αυτοκίνητο μετράει τιμές στο διάστημα [30,127]. Σε αυτήν την περίπτωση με 7 bits μπορεί να αναπαρασταθεί η θερμοκρασία αλλά οι πρώτες 29 τιμές δεν θα μπορούν να αναπαραστήσουν κάτι χρήσιμο.

Προβληματική αναπαράσταση ακεραίων

1. Αν $x \in [0, L]$ και το L δεν είναι δύναμη του 2 υπάρχουν προβλήματα ακρίβειας.
2. **Λύσεις**:
   1. Απεικονίζουμε τους αριθμούς στο [0,L-2] με δυαδική αναπαράσταση και το L-1 με άλλη αναπαράσταση. Δηλαδή ένας αριθμός μπορεί να πιάσει σημαντικό μέρος των bits σε σχέση με τους υπόλοιπους αριθμούς.
   2. Μετατροπή του αριθμού:

$$y(x) = \frac{x}{L-1}\left(2^N - 1\right)$$

, δηλαδή ουσιαστικά γίνεται καλύτερη κατανομή των bits

Παράδειγμα προβληματικής αναπαράστασης

1. Έστω ότι $x = 10$. Απαιτούνται 4 bits για την αναπαράσταση καθώς $2^4 = 16 > 10 > 2^3 = 8$.
2. Με το clipping απεικονίζεται ο αριθμός 10 με όλες τις αναπαραστάσεις μετά το 1010.
3. Χρησιμοποιούνται το 37.5% των αναπαραστάσεων για ένα μόνο αριθμό.
4. Με την δεύτερη λύση πιθανόν κάποιοι αριθμοί να έχουν δυο αναπαραστάσεις αντί για μια.

Αναπαράσταση δεκαδικών τιμών

1. **Δυαδική αναπαράσταση**: Χρήση δυαδικών τιμών για την αναπαράσταση των δεκαδικών τιμών.
2. **Άμεση αναπαράσταση**: Απευθείας με χρήση δεκαδικών τιμών.

Δυαδική αναπαράσταση δεκαδικών τιμών

1. Αν διαθέτουμε N δυαδικά στοιχεία και $x \in [x_{min}, x_{max}]$, τότε δύο διαδοχικές τιμές $x_1,\ x_2$ θα απέχουν μεταξύ τους

$$|x_2 - x_1| = \frac{2^N - 1}{x_{max} - x_{min}}$$

2. Παράδειγμα: N=10 και $x \in [-5, 5]$, τότε οι αριθμοί θα απέχουν μεταξύ τους 0.01
3. Τέτοια σφάλματα αναπαράστασης δεν είναι αποδεκτά σε εφαρμογές
4. Λύση: Μεγαλύτερα χρωμοσώματα. Κάνει πιο αργή την υλοποίηση αλλά το πρόβλημα ακρίβειας δεν εξαφανίζεται.

Άμεση αναπαράσταση δεκαδικών τιμών

1. Οι ίδιοι αριθμοί είναι και τα χρωμοσώματα
2. Η πιο κοινή λύση
3. Η ακρίβεια εξαρτάται μόνον από την ακρίβεια του υπολογιστή
4. Θα πρέπει να επινοηθούν νέοι γενετικοί τελεστές
5. Δεν υπάρχουν αποδείξεις σύγκλισης

Γενικά στοιχεία

1. Είναι ο θεμέλιος λίθος των γενετικών αλγορίθμων
2. Πρέπει να επικροτεί τα καλά χρωμοσώματα
3. Πρέπει να διακρίνει τα χρωμοσώματα μεταξύ τους
4. Μπορεί να δουλέψει για μεγιστοποίηση ή για ελαχιστοποίηση.

Γενική μορφή

```
function fitness(x)
 do_something
 compute value
 return value
end fitness
```

1. x είναι το χρωμόσωμα σε μορφή πίνακα
2. do_something: εξαρτάται από το πρόβλημα
3. value: είναι η τελική τιμή καταλληλότητας

Μέγιστος αριθμός μεταβάσεων

```
function fitness(x)
 value=0
 for i=1..size(x)-1 do
   if(x(i)=0 and x(i+1)=1) then
     value=value+1
   end if
 end for
 return value
end fitness
```

Ελαχιστοποίηση συναρτήσεων χωρίς περιορισμούς

1. Είναι το κύριο πεδίο των Γενετικών αλγορίθμων
2. Ελαχιστοποίηση της συνάρτησης: $f(x) : \ S \subset R^n \rightarrow R$
3. Αν κάνουμε ελαχιστοποίηση, τότε η καταλληλότητα $= f(x)$
4. Αν κάνουμε μεγιστοποίηση, τότε η καταλληλότητα $= -f(x)$

Ελαχιστοποίηση χωρίς περιορισμούς (παράδειγμα)

- Ελαχιστοποίηση της συνάρτησης
  $f(x) = x_1^2 + x_2^2,\ x \in [-10, 10]^2$

```
function fitness(x)
 if( x(1)>10 or x(1)<-10) return fail
 if( x(2)>10 or x(2)<-10) return fail
 return -(x(1)*x(1)+x(2)*x(2))
end fitness
```

Το πρόβλημα TSP

1. Υπάρχουν μια σειρά από πόλεις
2. Ένας πωλητής πρέπει να περάσει από κάθε πόλη
3. Μπορεί να επισκεφτεί τις πόλεις μόνο μια φορά
4. Για λόγους απλότητας θεωρούμε πως όλες οι πόλεις συνδέονται μεταξύ τους
5. Στις περισσότερες υλοποιήσεις υπάρχει και κόστος ανάμεσα στις πόλεις (πχ χιλιόμετρα)

Το πρόβλημα TSP (καταλληλότητα)

```
function fitness(x)
 value=0
 for i=1..size(x)-1 do
    value = value +distance (x(i),x(i+1))
 end for
 return value
end fitness
```

1. x είναι μια διαδρομή
2. dist(a,b) είναι η απόσταση ανάμεσα στην πόλη a και στην πόλη b

Γενικά στοιχεία

1. Χρησιμοποιείται για την δημιουργία της mating pool (λίστα χρωμοσωμάτων)
2. Η λίστα αυτή θα χρησιμοποιηθεί στην διασταύρωση
3. Είναι στοχαστική διαδικασία
4. Θεωρείται η βάση της εξέλιξης

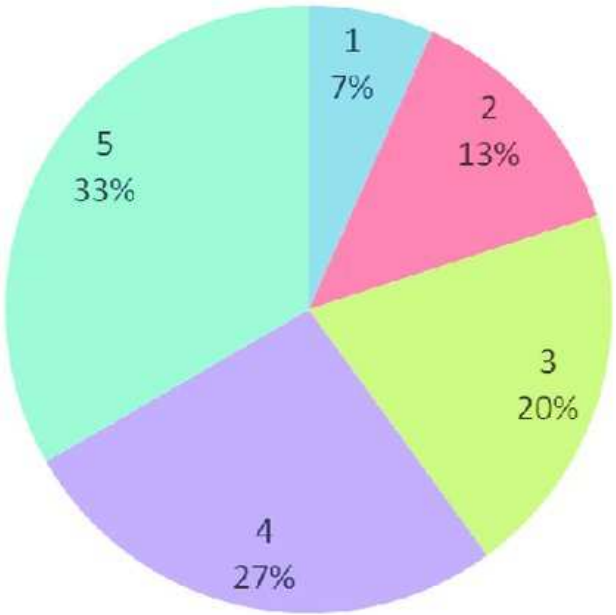
Επιλογή ρουλέττας

1. Χρησιμοποιείται πιο συχνά
2. Ανάλογη της καταλλήλοτητας
3. Η επιλογή γίνεται με πιθανότητα

$$p_i = \frac{f_i}{\sum_{j=1}^{M} f_j}$$

4. Θα πρέπει να είναι θετικές οι καταλληλότητες
5. Οπτικά παρουσιάζεται σαν ρουλέττα, όπου κάθε μέλος έχει πιθανότητα να επιλεγεί ανάλογα του χώρου που πιάνει στην ρουλέττα.

Ρουλέττα (σχήμα)



Individual	Fitness
1	1.0
2	2.0
3	3.0
4	4.0
5	5.0

Υλοποίηση ρουλέττας με πίνακα ακεραίων

```
function select(p,M)
 start=1
 for i=1..size(p) do
   k=p(i)*M
   for j=start..start+k do
     v(j)=i
   end for
   start=start+k+1
 end for
 return v ( rand(1,M) )
end select
```

1. Κάθε χρωμόσωμα πιάνει τόσες θέσεις στον πίνακα ανάλογα με την καταλληλότητά του
2. M είναι η διάσταση του πίνακα

Παράδειγμα

1. Έστω οι καταλληλότητες: $f_1 = 12,\ f_2 = 10,\ f_3 = 15,\ f_4 = 25$
2. Έστω M=12
3. Παράγεται ο πίνακας

$$v = [1, 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 4]$$

4. Πολύπλοκη υλοποίηση
5. Κρίσιμη η επιλογή του M
6. Ευνοούνται περισσότερα τα καλύτερα χρωμοσώματα

Υλοποίηση ρουλέττας με δεκαδικές τιμές

1. Γίνεται ταξινόμηση σε αύξουσα σειρά των καταλληλοτήτων
2. Επιλέγεται ένας τυχαίος στο διάστημα $r \in \left[0, \sum_{j=1}^{N} f_j\right)$
3. Επιλέγεται για αναπαραγωγή εκείνο το χρωμόσωμα με

$$\sum_{j=1}^{i-1} f_j \leq r \leq \sum_{j=1}^{i} f_j$$

Αλγόριθμος ρουλέττας με δεκαδικές τιμές

```
function select(f)
  totalf=0
  for i=1..size(f) do
   totalf=totalf+f(i)
  end for
  sum=0
  r=rand(0,totalf)
  for i=1..size(f) do
    sum=sum+f(i)
    if(sum>=r) return i
  end for
end select
```

Προβλήματα ρουλέττας

1. Δύσκολη υλοποίηση
2. Απαίτηση για ταξινόμηση καταλληλοτήτων
3. Αργή υλοποίηση
4. Πρόωρη σύγκλιση
5. Στασιμότητα
6. Επίλυση των παραπάνω με τεχνικές κλιμάκωσης

Τεχνικές κλιμάκωσης

1. Τροποποίηση καταλληλότητας
2. Δεν υπάρχει γενική μέθοδος κλιμάκωσης
3. Η κλιμάκωση εξαρτάται από το πρόβλημα

Κλιμάκωση (παράδειγμα 1)

1. Καταληλλότητες: 9.8, 10.0, 10.2, 11.0
2. Πολύ κοντά μεταξύ τους
3. Ισοπίθανη επιλογή
4. Προσθέτουμε στο καλύτερο τιμή M, στο επόμενο M/2 κτλ
5. Παράδειγμα για M=100: 22.3, 35.0, 60.2, 111
6. Αρκετά κρίσιμη η επιλογή του M
7. Δεν είναι απαραίτητο να διαιρούμε με το 2, γιατί μηδενίζεται πολύ γρήγορα.

Κλιμάκωση (παράδειγμα 2)

1. Καταλληλότητες: 10^{10}, 10^{20}, 10^{40}, 10^{45}
2. Πολύ μεγάλες τιμές
3. Πολύ μακριά μεταξύ τους
4. Λύση: χρήση λογαρίθμων: 10, 20, 40, 45

Κλιμάκωση (παράδειγμα 3)

1. Καταλληλότητες: 1.5, 1.6, 50.0, 100.0, 100.0
2. Υπάρχουν μαζί μικρές και μεγάλες τιμές
3. Λογαρίθμηση: 0.18, 0.20, 1.7, 2.0, 2.0

Κλιμάκωση στο [0,1]

1. Υπολογίζονται οι ποσότητες $f_{min},\ f_{max}$
2. Υπολογισμός νεας καταλληλότητας:

$$f_i^s = \frac{f_i - f_{min}}{f_{max} - f_{min}}$$

3. Παράδειγμα: $\{2, 3, 4, 5, 6\} \rightarrow \left\{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\right\}$

Επιλογή tournament

1. Η πιο απλή μέθοδος επιλογής
2. Επιλέγεται το καλύτερο από μια μικρή ομάδα πληθυσμού N ατόμων
3. Δεν χρειάζεται ταξινόμηση
4. Όλα τα χρωμοσώματα έχουν πιθανότητα να επιλεγούν
5. Είναι η μέθοδος που ακολουθεί η φύση
6. Κρίσιμη η επιλογή του N

Επιλογή tournament (αλγόριθμος)

```
function select(N)
 max_fitness=0
 parent = 0
 for i=1..N do
  r = rand(1,chromosome_count)
  if(fitness(r)>max_fitness) then
     max_fitness = fitness(r)
     parent = r
  end if
 end for
 return parent
end select
```

Ελιτίστικη επιλογή

1. Το καλύτερο ή τα M καλύτερα διατηρούνται ανέπαφα από γενιά σε γενιά
2. Είναι κυρίως μέθοδος βελτίωσης και όχι μέθοδος επιλογής
3. Υπάρχει εγγύηση πως μια καλή λύση δεν θα χαθεί
4. Μπορεί να προκαλέσει πρόωρη σύγκλιση για μεγάλες τιμές του M
5. Συνήθως M=1

Γενικά

1. Χρησιμοποιεί την επιλογή
2. Ανταλλαγή γενετικού υλικού
3. Υπάρχουν πολλοί τρόποι διασταύρωσης
4. Παράγονται νέα έτομα

Ρυθμός διασταύρωσης

1. Συμβολίζεται με p_c
2. Καθορίζει πόσα νέα μέλη θα παραχθούν
3. Τα νέα χρωμοσώματα συνήθως αντικαθιστούν παλαιότερα
4. Αν γίνεται έλεγχος για καλύτερες τιμές καταλληλότητας πριν την αντικατάσταση μιλάμε για steady state γενετικούς

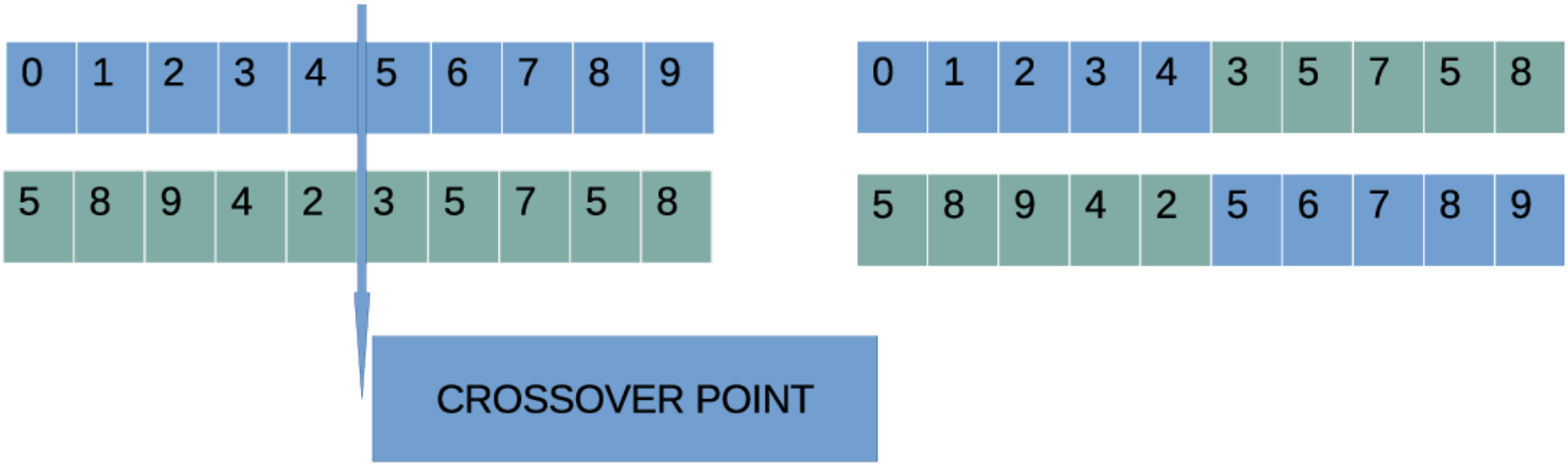
Steady state διασταύρωση

1. Τα νέα χρωμοσώματα διώχνουν μόνο τα χειρότερα
2. Σε πολλές περιπτώσεις μπορούν να συνυπάρχουν μαζί με τους γονείς αλλά με ανώτατο όριο στον πληθυσμό

Διασταύρωση ενός σημείου

1. Επιλέγεται ένα τυχαίο σημείο στο χρωμόσωμα
2. Γίνεται ανταλλαγή ανάμεσα σε δύο χρωμοσώματα με βάση αυτό το σημείο
3. Η πιο κοινή μέθοδος σε ακέραια χρωμοσώματα

Διασταύρωση ενός σημείου(σχήμα)



Διασταύρωση ενός σημείου (παράδειγμα)

1. Χρωμοσώματα: $x1 = [10, 5, 6]$ $x_2 = [1, 4, 9, 8, 12]$
2. Σημείο διασταύρωσης: 2
3. Απόγονοι: $c1 = [10, 5, 9, 8, 12]$, $c_2 = [1, 4, 6]$
4. Μπορεί να χρησιμοποιηθεί και σε μεταβλητού μήκους χρωμοσώματα

Ομοιόμορφη διασταύρωση

```
for i=1..chromosome_size do
  r = rand(0,1)
  if(r<=0.5) then
    children1(i)=parent1(i)
    children2(i)=parent2(i)
   else
    children1(i)=parent2(i)
    children2(i)=parent1(i)
  end if
end for
```

Δεκαδική διασταύρωση

```
for i=1..chromosome_size do
  children1(i)=parent1(i)+a*(parent1(i)-parent2(i))
  children2(i)=parent2(i)+a*(parent2(i)-parent1(i))
end for
```

1. Ο αριθμός a είναι στο διάστημα [-0.5,1.5]
2. Χρησιμοποιείται σε δεκαδικές αναπαραστάσεις κυρίως

Γενικά στοιχεία

1. Μικρές τυχαίες αλλαγές
2. Συμβαίνουν πολύ σπάνια
3. Εξάρτηση από τον ρυθμό μετάλλαξης $p_m \leq 1$
4. Συνήθως
$$p_m = \frac{1}{chromosomesize}$$
5. Συνήθως χρησιμοποιείται για να ανακτηθεί παλιό υλικό παρά για να βρεθεί καινούριο

Μετάλλαξη (αλγόριθμος)

```
for i=1..chromosome_count do
  for j=1..chromosome_size do
    r = rand(0,1)
    if(r<=pm) then
      change (chromosome(i), j)
    end if
  end for
end for
```

Η διαδικασία change() εξαρτάται από το πρόβλημα

Δεκαδική μετάλλαξη

```
for i=1..chromosome_size do
  r=rand(0,1)
  if(r<pm) then
    p=rand(0,1)
    if(p>0.5) then
      x(i) = x(i) +(xmax-x(i))*(1-r^(1-(t/T))*b)
     else
      x(i) = x(i) +(xlow-x(i))*(1-r^(1-(t/T))*b)
    end if
  end if
end for
```

Δεκαδική μετάλλαξη

1. $x_i \in [x_{low},\ x_{max}]$
2. t είναι ο αριθμός γενιάς
3. T είναι ο μέγιστος αριθμός γενιών
4. Η σταθερά b ελέγχει την διαφοροποίηση στο χρωμόσωμα. Συνήθως $b = 4$

Σύνοψη

- Γενικά στοιχεία
- Αναπαραστάσεις
- Καταλληλότητα
- Γενετικοί τελεστές

Βιβλιογραφία I

- Rosenblatt, Frank (1958), The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, Cornell Aeronautical Laboratory, Psychological Review, v65, No. 6, pp. 386–408.
- Freund, Y. and Schapire, R. E. 1998. Large margin classification using the perceptron algorithm. In Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT' 98). ACM Press.