

# Τεχνητή Νοημοσύνη

## Μεθευρετικές Μέθοδοι

Δρ. Δημήτριος Κουτσομητρόπουλος

1

## Αλγόριθμοι αναζήτησης λύσης

### (Μεθ-) Ευρετικές Μέθοδοι (Metaheuristics)

- ▶ **Hill Climbing, Local Search** Αναρρίχηση Λόφων ή Τοπική αναζήτηση
- ▶ **Random-restart** Τυχαία Επανεκκίνηση
- ▶ **Iterative Local Search** Επαναληπτική Τοπική Αναζήτηση
- ▶ **Local Beam Search** Τοπική Ακτινική Αναζήτηση
- ▶ **Αναζήτηση Tabu**
- ▶ **Simulated Annealing** Προσομοιωμένη Ανόπτηση
- ▶ **Γενετικοί Αλγόριθμοι**
- ▶ **Αναζήτηση Tabu**
- ▶ **PSO, ACO, BCO, ...**

▶ 2

2

## Μεθευρετικές μέθοδοι

- ▶ Πρότυπες μεθοδολογίες (templates) για επίλυση γενικευμένων κατηγοριών προβλημάτων, ιδίως βελτιστοποίησης
  - ▶ δεν εξαρτώνται από το εκάστοτε πρόβλημα
- ▶ Χρησιμοποιούν μια τοπική ευρετική για την δημιουργία νέων λύσεων
  - ▶ εξαρτάται από το πρόβλημα
- ▶ Δεν είναι ούτε συνεπείς ούτε πλήρεις
  - ▶ μπορούν να εντοπίσουν μια ικανοποιητική λύση σε σύντομο χρόνο
- ▶ Μπορεί να είναι:
  - ▶ *κατασκευαστικοί ή άπληστοι*, χτίζοντας σταδιακά τη λύση σε κάθε βήμα (βλ. BestFS)
  - ▶ *επαναληπτικοί*, που ξεκινούν από μια πλήρη λύση (ή πληθυσμό) και τη βελτιώνουν σταδιακά σε κάθε βήμα
    - ▶ έχουν ως βάση την **τοπική αναζήτηση (local search)**

▶ 33

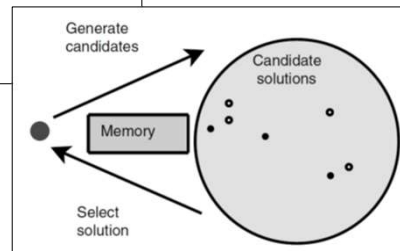
3

## Τοπική Αναζήτηση ή Αναρρίχηση Λόφων (Hill Climbing)

- ▶ Υπάρχει μια **αντικειμενική συνάρτηση** (objective function)  $f: S \rightarrow R$ , που θέλουμε να βελτιστοποιήσουμε
  - ▶ Συνάρτηση καταλληλότητας (fitness function)
  - ▶ Συνάρτηση κόστους (cost function)

```
s = s0 ; /* Generate an initial solution s0 */
While not Termination_Criterion Do
  Generate (N(s)) ; /* Generation of candidate neighbors */
  If there is no better neighbor Then Stop ;
  s = s' ; /* Select a better neighbor s' ∈ N(s) */
Endwhile
Output Final solution found (local optima).
```

- ▶ Τί γίνεται αν οι καλύτεροι γείτονες έχουν την ίδια τιμή με τον τρέχοντα;
  - ▶ **Πλατό** (plateau): Μια επίπεδη περιοχή στη συνάρτηση – επίπεδο τοπικό βέλτιστο
  - ▶ Μπορούμε να επιτρέψουμε στον αλγόριθμο να συνεχίσει, για κάποια βήματα



▶ 4

4

## Τοπική Αναζήτηση: Η έννοια της γειτονιάς

- ▶ Γειτονιά της  $s$ 
  - ▶ Εξαρτάται από την αναπαράσταση
  - ▶ Τοπικότητα (locality): μικρές αλλαγές στην αναπαράσταση επιφέρουν μικρές αλλαγές στη λύση (αλλιώς *τυχαία αναζήτηση*)
  - ▶ Συνδετικότητα (connexity): Μπορούμε να φτάσουμε από μία κατάσταση σε οποιαδήποτε άλλη στο χώρο καταστάσεων
  - ▶ Τι μέγεθος/ακτίνα έχει; Μικρό μέγεθος για γρήγορη εξερεύνηση, μεγάλο για αποφυγή **τοπικών βέλτιστων**

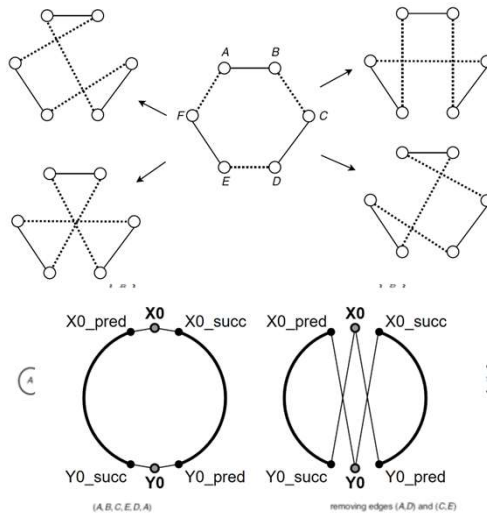


▶ 5

5

## Τοπική Αναζήτηση: Τελεστές μετάβασης

- ▶ Πώς προκύπτουν νέες λύσεις;
  - ▶ Τι τελεστές μετάβασης εφαρμόζονται;
  - ▶ Τοπικές ευρετικές (π.χ. TSP: swap, 2-opt, k-opt)

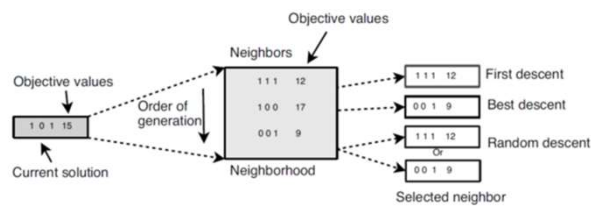


▶ 6

6

## Τοπική Αναζήτηση: αρχική λύση και επιλογή βελτίωσης

- ▶ Πώς φτιάχνεται η αρχική λύση;
  - ▶ Τυχαία επιλογή
  - ▶ Άπληστος αλγόριθμος
  - ▶ Η άπληστη προσέγγιση δεν είναι αναγκαστικά καλύτερη  
 Στο TSP η ευρετική Clarke-Wright δίνει καλύτερη λύση από μια τυχαία επιλογή. Όμως η τοπική αναζήτηση θα οδηγηθεί σε χειρότερες λύσεις!
- ▶ Πώς επιλέγεται η  $s'$ ;
  - ▶ Είναι η **καλύτερη** που βρέθηκε; (TSP: εφαρμογή όλων των 2-opt κινήσεων)
  - ▶ Είναι η **πρώτη καλύτερη** που βρέθηκε; (εφαρμογή 2-opt διαδοχικά μέχρι να βρεθεί καλύτερη) – δίνει την ίδια ποιότητα λύσεων στην πράξη!
  - ▶ Είναι **τυχαία επιλογή** από τις καλύτερες;



▶ 7

7

## Παράδειγμα: 8 βασίλισσες

- ▶ Κάθε βασίλισσα σε διαφορετική στήλη,  $8^8=17M$  καταστάσεις
- ▶ Αναπαράσταση κατάστασης ως διάνυσμα 8 θέσεων
- ▶ Τελεστής μετάβασης: **Μεταβολή μίας** θέσης στο διάστημα [1, 8]
  - ▶ Μέγεθος γειτονιάς:  $8 \times 7 = 56$  ( $7+7+\dots$ )

18	12	14	13	13	12	14	14
14	16	13	15	12	14	12	16
14	12	18	13	15	12	14	14
15	14	14	♠	13	16	13	16
♠	14	17	15	♠	14	16	16
17	♠	16	18	15	♠	15	♠
18	14	♠	15	15	14	♠	16
14	14	13	17	12	14	12	18

<4, 3, 2, 5, 4, 3, 2, 3>,  $f = 17$

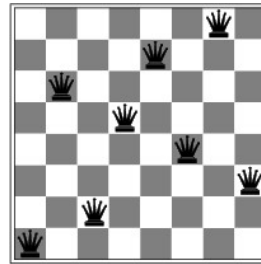
- ▶ **Συνάρτηση αξιολόγησης:** Ο αριθμός των βασιλισσών που μπορεί να επιτίθενται η μια στην άλλη
- ▶ Ο καλύτερος γείτονας έχει κόστος 12

▶ 8

8

## Παράδειγμα: 8 βασίλισσες – τοπικό ελάχιστο

- ▶ Νέα κατάσταση με  $f=1$  σε 5 κινήσεις
- ▶ Όμως, **τοπικό ελάχιστο!**
  - ▶ Οποιαδήποτε μετάβαση δίνει χειρότερη λύση
  - ▶ Ο αλγόριθμος τερματίζει
  - ▶ 86% τοπικό ελάχιστο (σε 3 κινήσεις μ.ο.)
  - ▶ 14% ολικό ελάχιστο ( $f=0$ , σε 4 κινήσεις μ.ο.)



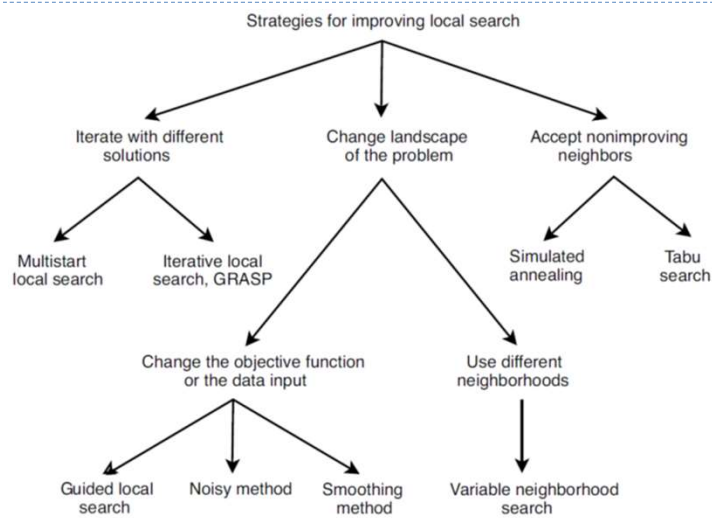
$\langle 1, 6, 2, 5, 7, 4, 8, 3 \rangle, f = 1$

- ▶ Επιτρέπουμε κινήσεις σε πλατό;
  - ▶ Αν ναι, μπορεί να προκύψει ατέρμονος βρόχος
  - ▶ Όριο 100 κινήσεων: Ποσοστό επιτυχίας 94%, αλλά 21 ή 64 μ.ο. κινήσεων επιτυχίας/αποτυχίας

▶ 9

9

## Βελτιώσεις στην τοπική αναζήτηση

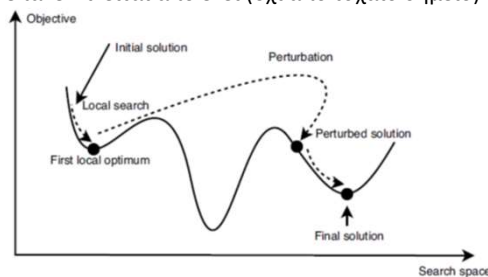


▶ 10

10

## Τυχαία επανεκκίνηση (Random-restart)

- ▶ Random restart ή *multistart* local search
  - ▶ Σε αποτυχία, η αναζήτηση δε σταματά, αλλά επανεκκινείται από τυχαίο σημείο
  - ▶ Σε ορισμένα προβλήματα απαιτούνται πάρα πολλές τέτοιες επανεκκινήσεις μέχρι να βρεθεί (ικανοποιητική) λύση (π.χ. διχοτόμηση γράφου)
  - ▶ Ο αριθμός τους αυξάνει εκθετικά με το μέγεθος του προβλήματος
  - ▶ Κεντρικό οριακό φαινόμενο:  
*Όταν το μέγεθος του προβλήματος γίνει πολύ μεγάλο, τα τοπικά ελάχιστα που προκύπτουν από τυχαίες αρχικές λύσεις είναι παρεμφερή ως προς την ποιότητά τους*
- ▶ Iterated local search (ILS)
  - ▶ Εφαρμόζεται μια **διατάραξη** (perturbation) στο τοπικό ελάχιστο και η αναζήτηση επανεκκινείται από εκεί (όχι από τυχαίο σημείο)



▶ 11

11

## Τοπική ακτινική αναζήτηση (Local Beam Search)

- ▶ Αποθηκεύονται  $k$  διαφορετικές καταστάσεις, αντί για μόνο μία
  - ▶ Αρχικά  $k$  καταστάσεις επιλέγονται τυχαία
  - ▶ Εκτελείται 1 βήμα local search για κάθε μία
  - ▶ Δημιουργείται η γειτονιά τους
  - ▶ Επιλέγονται οι  $k$  καλύτεροι γείτονες *από όλες τις γειτονιές*
  - ▶ Συνεχίζεται local search για καθέναν από αυτούς
  - ▶ Σύγκριση με random-restart?
- ▶ Συχνά οι  $k$  καλύτεροι τείνουν να συγκεντρώνονται στην ίδια περιοχή (τοπικό ελάχιστο)
  - ▶ Απορρίπτονται νωρίς χειρότερες λύσεις που θα μπορούσαν να οδηγήσουν σε βελτίωση αργότερα
  - ▶ Εκμετάλλευση vs. εξερεύνηση
- ▶ Stochastic local beam search:  $k$  γείτονες επιλέγονται με πιθανότητα ανάλογη της καταλληλότητας

▶ 12

12

## Αναζήτηση ταμπού (Tabu Search)

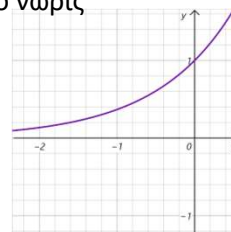
- ▶ Ύπαρξη μνήμης για αποθήκευση πληροφοριών σχετικών με τη διαδικασία αναζήτησης
- ▶ Όπως στην τοπική αναζήτηση, επιλέγεται ο καλύτερος γείτονας
- ▶ Τι γίνεται σε **τοπικό βέλτιστο**;
  - ▶ Η αναζήτηση συνεχίζει επιλέγοντας **χειρότερο** σημείο, δημιουργείται νέα γειτονιά, και επιλέγεται η καλύτερη λύση, ακόμα κι αν είναι χειρότερη από το τοπικό βέλτιστο
  - ▶ Μπορεί να δημιουργηθούν κύκλοι (επίσκεψη καταστάσεων που έχουν ήδη δημιουργηθεί)
- ▶ **Λίστα Ταμπού (Tabu list)**: Αποθηκεύονται οι λύσεις (ή οι τελεστές) που έχουν ήδη ελεγχθεί
  - ▶ Οι  $k$  πιο πρόσφατες
  - ▶ Όχι ολόκληρες οι λύσεις, αλλά χαρακτηριστικά τους (attributes)
  - ▶ Π.χ. μια λίστα ακμών για το TSP

▶ 13

13

## Προσομειωμένη Ανόπτηση (Simulated Annealing)

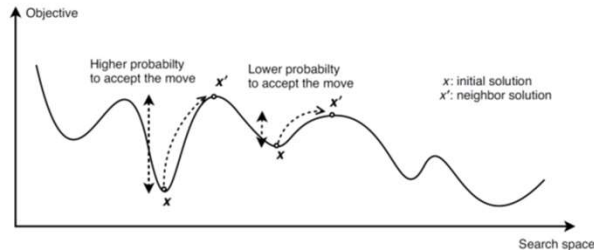
- ▶ Εμπνευσμένη από τη μεταλλουργία
  - ▶ **Ανόπτηση**: Θέρμανση σε **υψηλή θερμοκρασία** και μετά **σταδιακή ψύξη** με συγκεκριμένο ρυθμό, ώστε το υλικό να αποκτήσει βέλτιστες ιδιότητες (σκλήρυνση)
- ▶ Χειρότερες λύσεις μπορεί να γίνουν αποδεκτές από νωρίς
  - ▶ Επιλέγεται τυχαία μια γειτονική λύση
  - ▶ Αν είναι καλύτερη θα γίνει αποδεκτή
  - ▶ Αν είναι χειρότερη ( $f(x') > f(x)$ ), θα γίνει αποδεκτή με πιθανότητα  $p$ :
$$p = e^{-\frac{f(x') - f(x)}{T}}$$
    - ▶  $x'$ : γείτονας,  $T$ : θερμοκρασία
  - ▶ Όσο πιο μεγάλο το  $T$ , τόσο πιο μικρός ο εκθέτης και άρα πιο κοντά στη μονάδα (αφού είναι πάντα αρνητικός).
    - ▶ Χειρότερες λύσεις θα επιλέγονται αρκετά συχνά
  - ▶ Όσο μικραίνει το  $T$ , μεγαλώνει ο εκθέτης και απομακρυνόμαστε από την μονάδα προς το 0.
    - ▶ Χειρότερες λύσεις θα γίνουν πιο σπάνια αποδεκτές



▶ 14

14

## Προσομιωμένη Ανόπτηση (Simulated Annealing)



- ▶ Σύγκλιση στη βέλτιστη λύση!
  - ▶ Αν η θερμοκρασία μειώνεται αρκετά αργά, ο αλγόριθμος θα βρει το **ολικό βέλτιστο** με πιθανότητα  $\rightarrow 1$
  - ▶ Όμως, *ασυμπτωτικά* (σύγκλιση μετά από άπειρα βήματα)
- ▶ Επιλογές παραμέτρων
  - ▶ Τι αρχική θερμοκρασία;
  - ▶ Τι πρόγραμμα ανόπτησης (ψύξης);
- ▶ Πολύ καλά (near optimal) αποτελέσματα για TSP, VLSI

▶ 15

15

## Γενετικοί Αλγόριθμοι

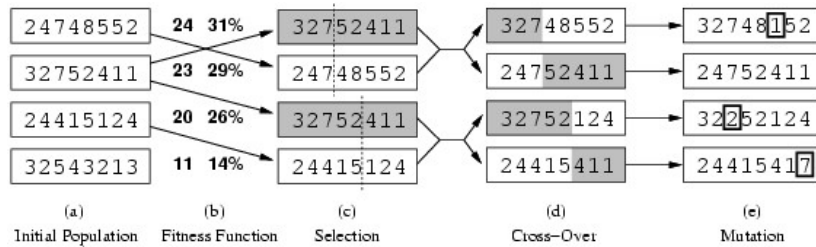
- ▶ Εμπνευσμένοι από τη θεωρία της εξέλιξης και της φυσικής επιλογής
  - ▶ *οι οργανισμοί που είναι περισσότερο προσαρμοσμένοι στο περιβάλλον τους επιβιώνουν και αναπαράγονται περισσότερο από τους λιγότερο προσαρμοσμένους*
- ▶ Μια διάδοχη κατάσταση προκύπτει συνδυάζοντας δύο καταστάσεις-γονείς
- ▶ Ξεκινά με  $k$  τυχαία επιλεγμένες καταστάσεις (**πληθυσμός**)
- ▶ Μια κατάσταση αναπαρίσταται ως *συμβολοσειρά* (0 και 1, ακέραιοι αριθμοί, κτλ)
- ▶ **Συνάρτηση καταλληλότητας**: Καλύτερες τιμές για καλύτερες λύσεις.
- ▶ Παραγωγή της επόμενης γενιάς χρησιμοποιώντας γενετικούς τελεστές
  - ▶ **Επιλογή** (selection)
  - ▶ **Διασταύρωση** (crossover)
  - ▶ **Μετάλλαξη** (mutation)
- ▶ Οι τελεστές εφαρμόζονται *στοχαστικά*

▶ 16

16



## Γενετικοί Αλγόριθμοι

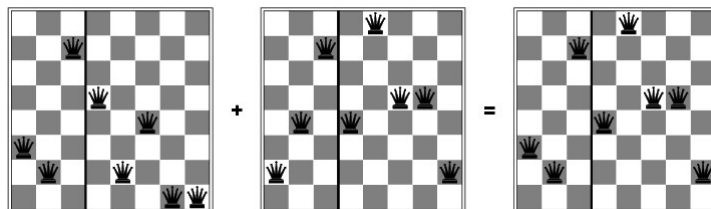


- ▶ Συνάρτηση καταλληλότητας: αριθμός ζευγαριών που δεν επιτίθενται ( $\min = 0$ ,  $\max = 8 \times 7/2 = 28$ )
- ▶ Επιλογή ανάλογη της απόδοσης (fitness proportionate selection):
  - ▶  $24/(24+23+20+11) = 31\%$
  - ▶  $23/(24+23+20+11) = 29\% \dots$

▶ 17

17

## Παράδειγμα διασταύρωσης



▶ 18

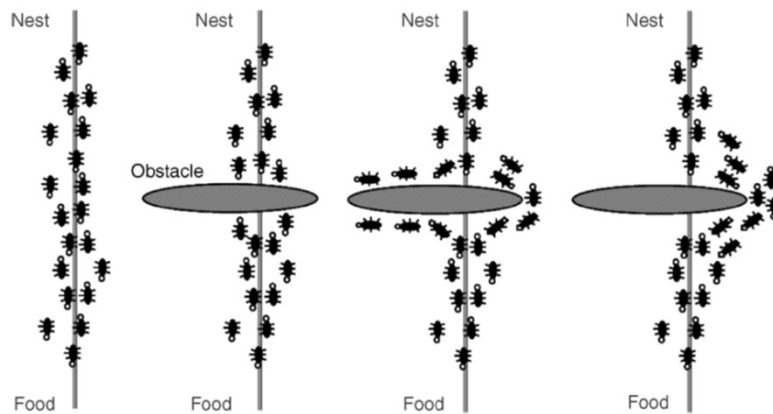
18

## Άλλες μέθοδοι

19

19

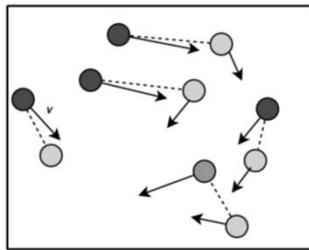
## Ant Colony Optimization (ACO) - 1992



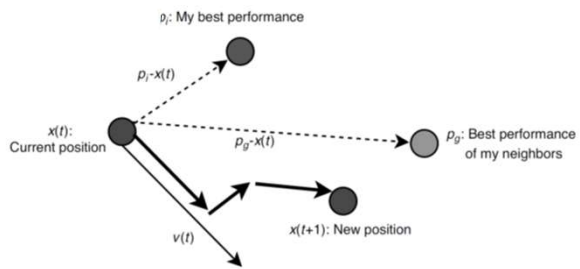
▶ 20

20

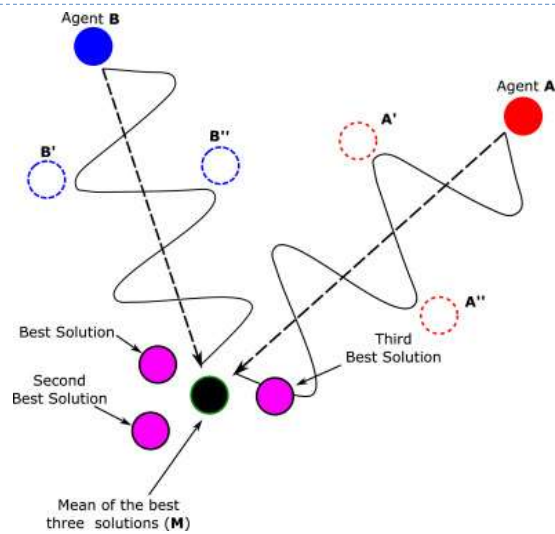
## Particle Swarm Optimization (PSO) - 2001



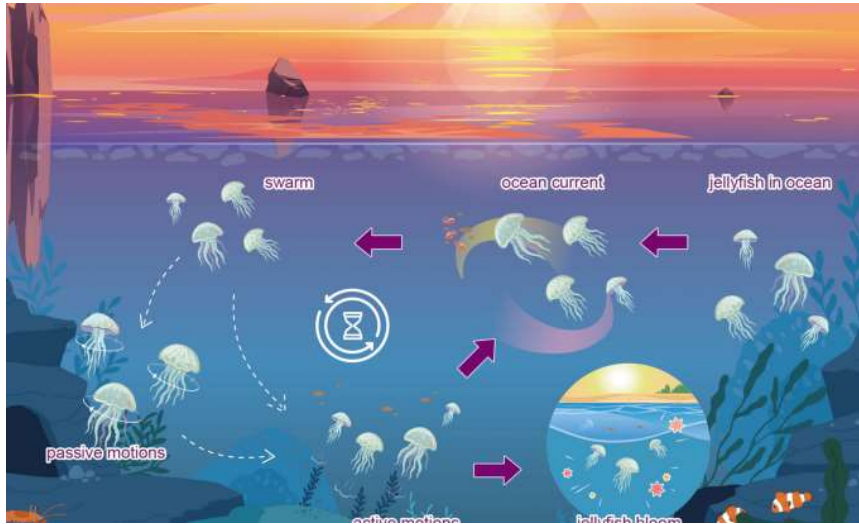
Decision space



## Social Ski Driver (SSD) - 2019



## Jellyfish Swarm Optimization (JSO) - 2022



▶ 23

23

## No free lunch theorem

- ▶ There's no such thing as free lunch!

«Δύο οποιοδήποτε αλγόριθμοι βελτιστοποίησης είναι ισοδύναμοι ως προς τη μέση επίδοσή τους σε όλα τα πιθανά προβλήματα»

Wolpert&McReady, 2005



- ▶ Δεν είναι όλες οι μέθοδοι βέλτιστες σε όλα τα προβλήματα

▶ 24 Πηγές: G. Talbi "Metaheuristics", S. Russell, P. Norvig "Artificial Intelligence: A modern Approach" 3d ed.

24