Last class

- Binary representation
- Integer representation
- Real-valued representation
- Permutation representation
- Tree representation

Representation
Mutation
Recombination

Heuristic Search and Evolutionary Algorithms

Lecture 7: Evolutionary Algorithms – Fitness, Selection and Population Management

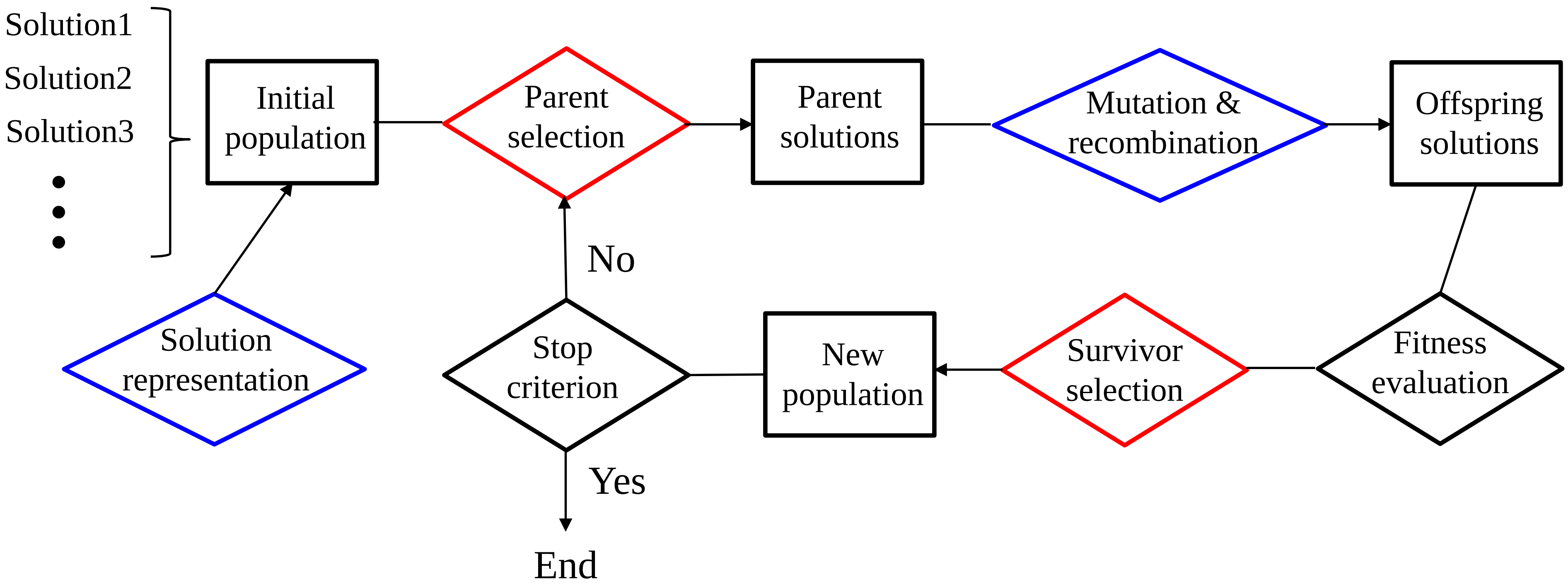
Chao Qian （钱超）

Associate Professor, Nanjing University, China

Email: qianc@nju.edu.cn
Homepage: http://www.lamda.nju.edu.cn/qianc/

Evolutionary algorithms

EAs share a common routine for $\arg\max_x f(x)$



Selection is independent of the solution representation, but based on the fitness

Parent selection: Fitness proportional selection

- Fitness proportional selection (FPS): the probability of selecting the i-th individual is

$$P_{FPS}(i) = \frac{f_i}{\sum_{j=1}^{\mu} f_j}$$

f_i: the fitness of the i-th individual, assumed to be non-negative

μ: the population size

- When fitness values are all very close, there is almost no selection pressure

Individual	Fitness	Sel. prob. P_{FPS}
A	101	0.326
B	104	0.335
C	105	0.339

⇒ Windowing:

$$f_i = f_i - \beta^t$$

e.g., $\beta^t = \min_{j \in \{1,2,\ldots,u\}} f_j$

the least fitness of the current population

Parent selection: Ranking selection

- Ranking selection (RS): the selection probabilities are based on relative rather than absolute fitness
  - Rank the individuals in the population from $\mu - 1$ (best) to 0 (worst) according to fitness

- Linear ranking selection (LRS):

rank ← i

$$\mathrm{P}_{LRS}(i) = \frac{2-s}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

The sum of the probabilities:

$$\sum_{i=0}^{\mu-1} \mathrm{P}_{LRS}(i) = \frac{2-s}{\mu} \cdot \mu + \frac{2(s-1)}{\mu(\mu-1)} \cdot \frac{\mu(\mu-1)}{2} = 1$$

Parent selection: Ranking selection

- Ranking selection (RS): the selection probabilities are based on relative rather than absolute fitness
  - Rank the individuals in the population from $\mu - 1$ (best) to 0 (worst) according to fitness

- Linear ranking selection (LRS):

rank

$$\mathrm{P}_{LRS}(i) = \frac{2-s}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

the probability of selecting the worst individual

Parent selection: Ranking selection

- Ranking selection (RS): the selection probabilities are based on relative rather than absolute fitness
  - Rank the individuals in the population from $\mu - 1$ (best) to 0 (worst) according to fitness

- Linear ranking selection (LRS):

rank

$$P_{LRS}(i) = \frac{2 - s}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

$s \in (1,2]$: the expected number of selecting the best individual after performing LRS for μ times

Parent selection: Ranking selection

- Linear ranking selection (LRS):

$$\mathrm{P}_{LRS}(i) = \frac{2-s}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}$$

The influence of $s \in (1,2]$: $\big(2i-(u-1)\big)\cdot s$

- As s increases, the prob. of selecting individuals with above-median fitness increases, while that with below-median fitness decreases
- When μ is odd, the probability of selecting the individual with median fitness is a constant, i.e., $1/\mu$

Individual	Fitness	Rank	P_{LRS} with $s = 1.5$	P_{LRS} with $s = 2$
A	1	0	0.1	0
B	4	1	0.15	0.1
C	5	2	0.2	0.2
D	7	3	0.25	0.3
E	9	4	0.3	0.4

Parent selection: Ranking selection

- Ranking selection (RS): the selection probabilities are based on relative rather than absolute fitness
  - Rank the individuals in the population from $\mu - 1$ (best) to 0 (worst) according to fitness

- Exponential ranking selection (ERS):

rank

$$\mathrm{P}_{ERS}(i) = \frac{1 - e^{-i}}{c}$$

normalization factor

$$\sum_{i=0}^{\mu-1} \frac{1 - e^{-i}}{c} = \frac{\mu - \frac{1 - e^{-\mu}}{1 - e^{-1}}}{c} = 1 \quad \Rightarrow \quad c = \mu - \frac{1 - e^{-\mu}}{1 - e^{-1}}$$

Implementation of sampling

- Roulette wheel

The probability of selecting the j-th individual

$$a_i = \sum_{j=1}^{i} \mathrm{P}_{sel}(j)$$

current_member = 1;

While current_member ≤ λ Do

  Pick a random value r uniformly from [0,1];

  $i = 1;$

  While $a_i < r$ Do

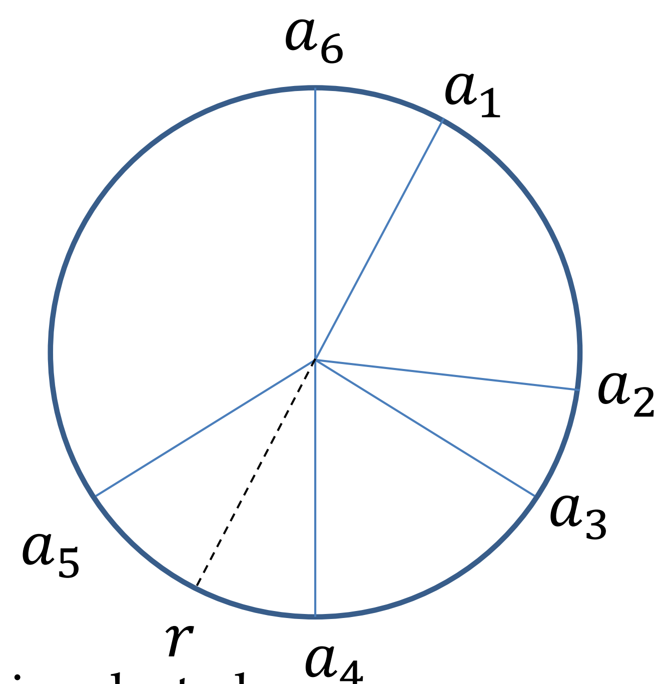
    $i = i + 1;$

  End While

  mating_pool[current_member] = parents[i];

  current_member = current_member + 1;

End While



The 5-th individual is selected

Implementation of sampling

- Roulette wheel

The expected number of the j-th individual in the mating pool

$$\lambda \cdot \mathrm{P}_{sel}(j)$$

The actual number of the j-th individual in the mating pool can be quite different from $\lambda \cdot \mathrm{P}_{sel}(j)$

How to make the actual number of the j-th individual in the mating pool close to $\lambda \cdot \mathrm{P}_{sel}(j)$?

Implementation of sampling

- Stochastic universal sampling

current_member = i = 1;

Pick a random value r uniformly from $[0,1/\lambda]$;

While current_member ≤ λ Do

    While $r \le a_i$ Do

        mating_pool[current_member] = parents[i];

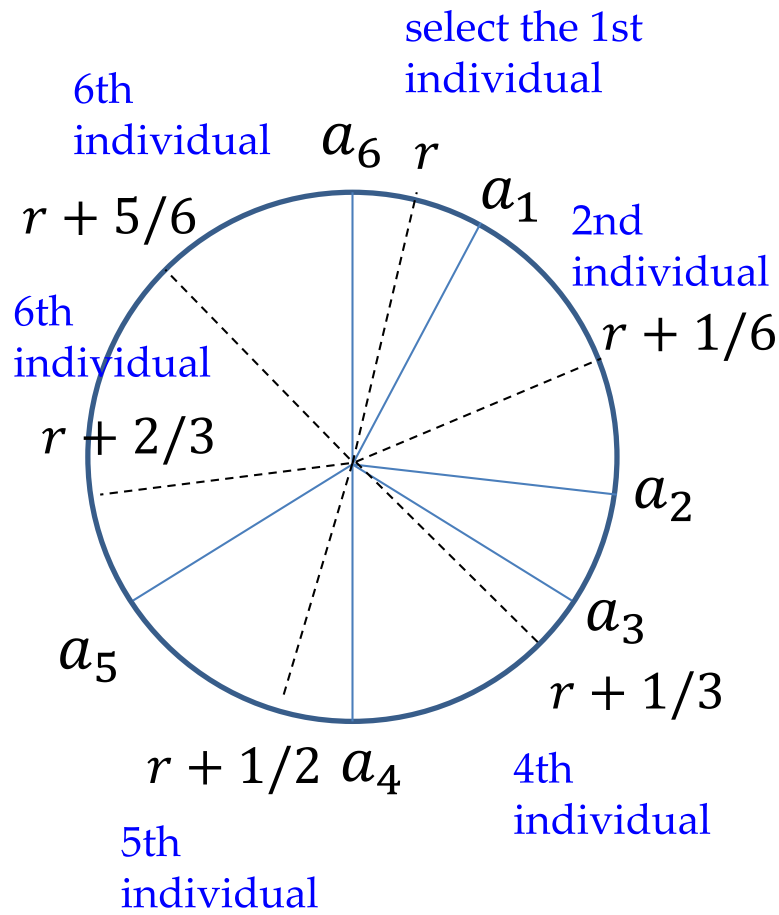
        $r = r + 1/\lambda$;

        current_member = current_member + 1;

    End While

    $i = i + 1$;

End While



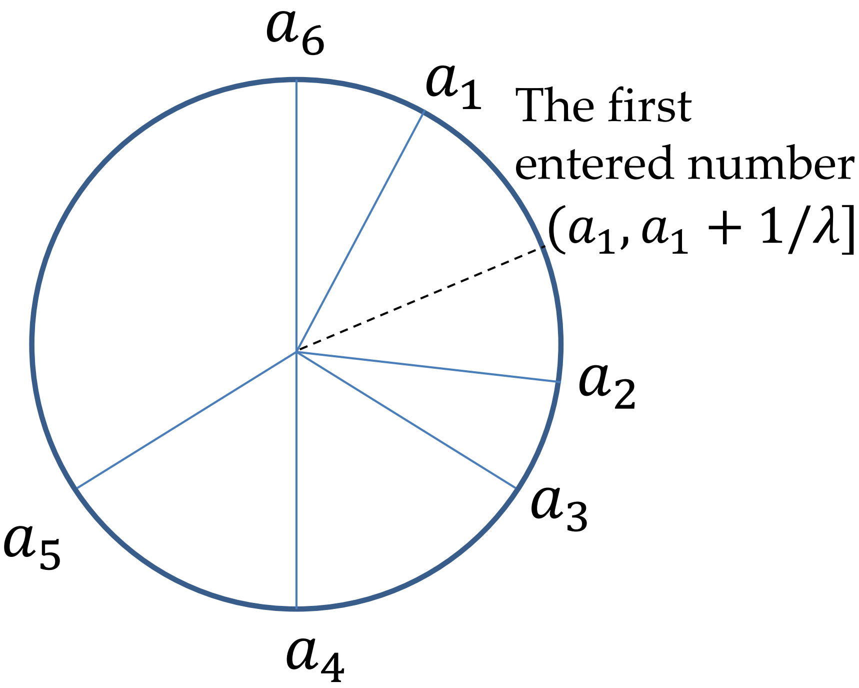
Implementation of sampling

- Stochastic universal sampling

The actual number of the j-th individual in the mating pool:

If $\lambda \cdot \mathrm{P}_{sel}(j)$ is an integer,
it must be $\lambda \cdot \mathrm{P}_{sel}(j)$

Otherwise, it must be
$\lfloor \lambda \cdot \mathrm{P}_{sel}(j) \rfloor$ or $\lfloor \lambda \cdot \mathrm{P}_{sel}(j) \rfloor + 1$



Parent selection: Tournament selection

- Tournament selection (TS): use only local fitness information
  - Pick k individuals randomly, with or without replacement;
  - Compare these k individuals, and select the best;
  - Repeat the above process for λ times independently

Assume that the selection is without replacement, and the best solution is unique

The probability of selecting the best solution at least once:

$$1-\left(1-\left(\binom{\mu-1}{k-1}\Big/\binom{\mu}{k}\right)\right)^{\lambda}=1-\left(1-(k/\mu)\right)^{\lambda}$$

Parent selection: Uniform selection

- Uniform selection (US): select an individual from the current population uniformly at random

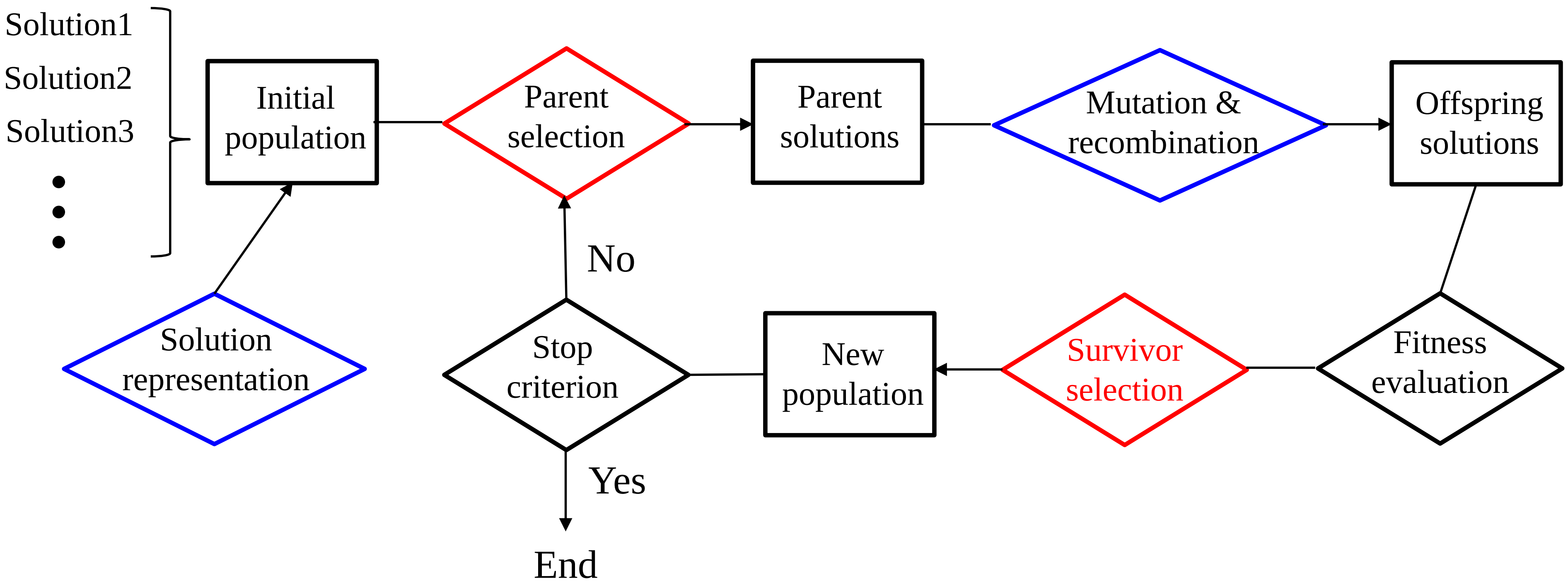
$$P_{US}(i) = \frac{1}{\mu}$$

The expected number of the j-th individual in the mating pool after performing uniform selection λ times

$$\lambda \cdot \frac{1}{\mu}$$

Evolutionary algorithms

EAs share a common routine for $\arg\max_{x} f(x)$



Selection is independent of the solution representation, but based on the fitness

Survivor selection

- Survivor selection: Manage the process of reducing the working memory of the EA from the current population and a set of λ offspring to a set of μ individuals forming the next population

  - Age-based replacement: fitness is not taken into account
  - Fitness-based replacement

Survivor selection: Age-based replacement

- Age-based replacement
  - Fitness is not taken into account
  - Each individual exists in the population for the same number of iterations

- For example, population size: μ, number of offspring: λ
  - If $\lambda = \mu$, the μ individuals in the current population are simply discarded, and replaced by the μ offspring
  - If $\lambda < \mu$, λ individuals (selected by the FIFO strategy) in the current population are replaced by the λ offspring

Survivor selection: Fitness-based replacement

Assume population size: μ, number of offspring: λ

- Replace worst (GENITOR) for $\mu > \lambda$
  - ➢ The worst λ individuals in the current population is replaced by the λ offspring
- (μ, λ) selection for $\mu < \lambda$

  May be better in leaving local optima

  - ➢ The best μ offspring forms the next population
- $(\mu + \lambda)$ selection
  - ➢ The best μ individuals from the current population and the λ offspring forms the next population

Survivor selection: Fitness-based replacement

Assume population size: μ, number of offspring: λ

- Round-robin tournament
  - Each individual x is evaluated against q other individuals randomly chosen from the current population and the offspring
  - For each comparison, a "win" is assigned if x is better than its opponent
  - The μ individuals with the greatest number of wins are retained to form the next population

The parameter q controls the selection pressure

Positively corelated

$q = \mu + \lambda - 1$ ⇨ $(\mu + \lambda)$ selection

$q = 1$ ⇨ Even the worst individual can be selected

Survivor selection

- Age-based replacement
- Fitness-based replacement
  - Replace worst
  - (μ, λ) selection
  - $(\mu + \lambda)$ selection
  - Round-robin tournament
- Parent selection mechanisms
  - Fitness proportional selection
  - Ranking selection
  - Tournament selection
  - Uniform selection

Survivor selection

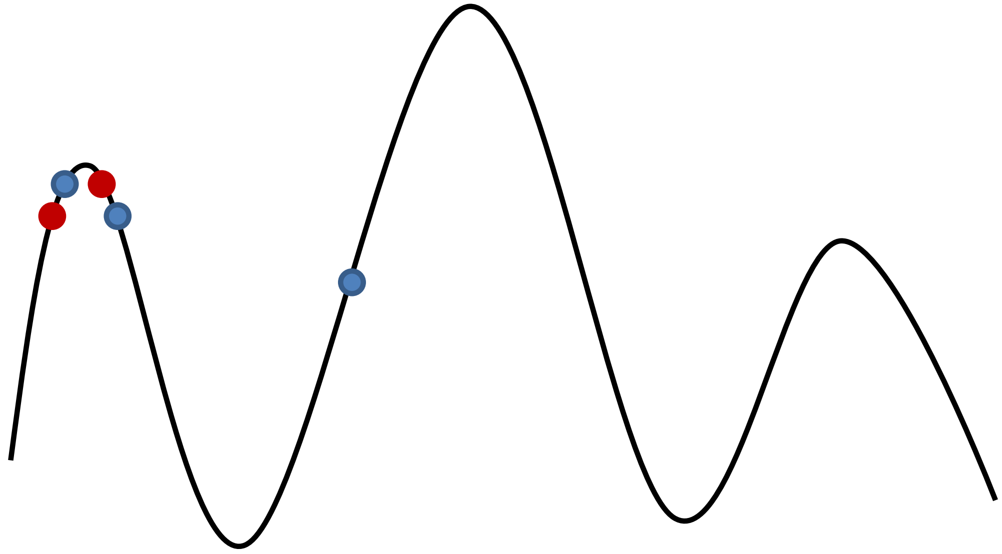
Elitism: the best generated individual is kept

Keep the best generated individual

Non-elitism

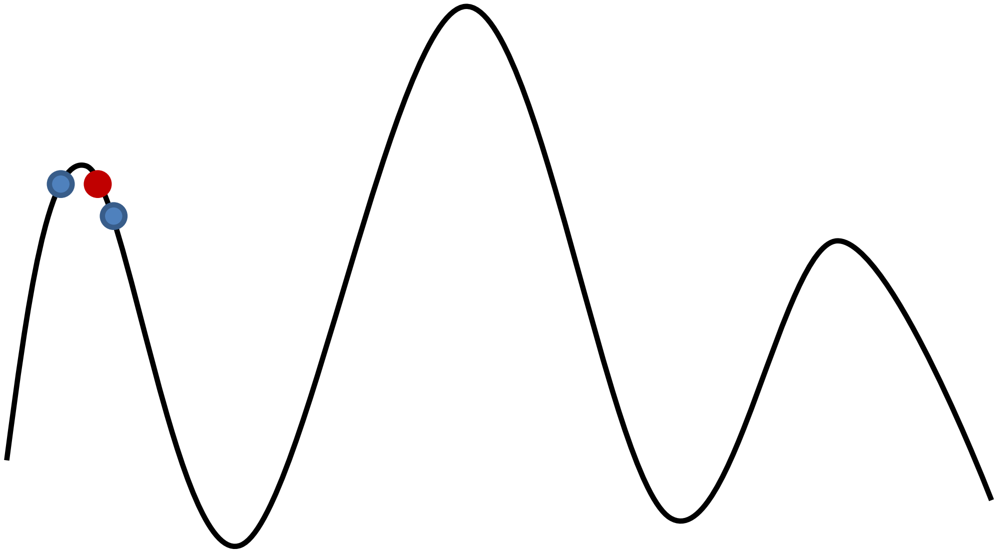
Population diversity

- Parent and survivor selection will make the EAs concentrate on one niche



Population diversity

- Parent and survivor selection will make the EAs concentrate on one niche

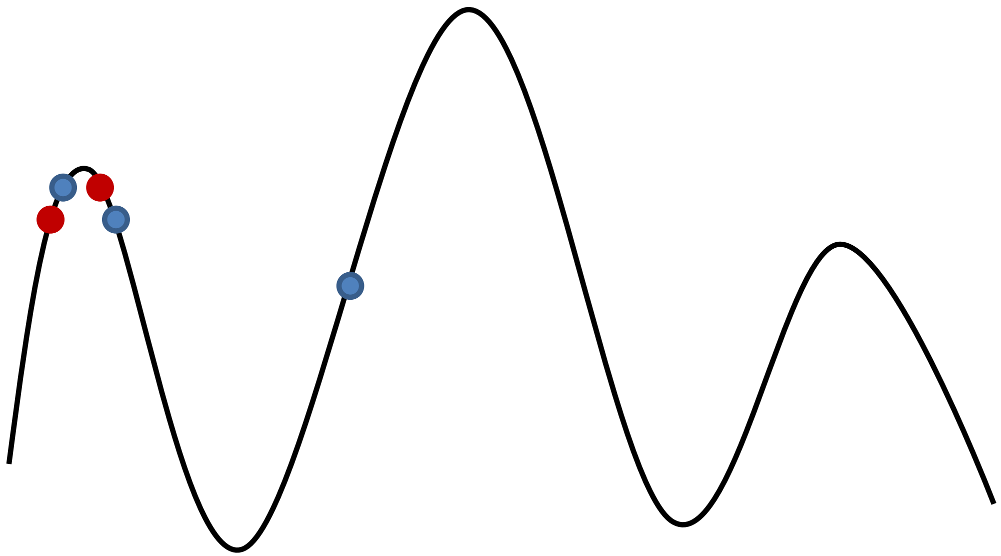


How to preserve sufficient diversity of the population?

Preserving diversity: Fitness sharing

- Fitness sharing: restrict the number of individuals within a niche by "sharing" their fitness

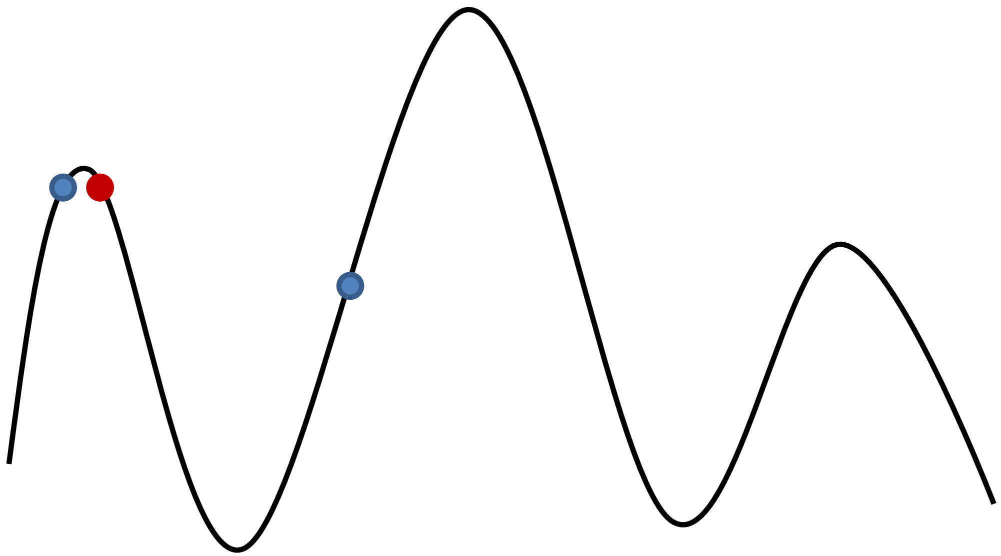
$$f'(i) = \frac{f(i)}{\sum_j sh(d(i,j))} \qquad sh(d) = \begin{cases} 1 - \left(\frac{d}{\delta_{share}}\right)^{\alpha} & if\ d \le \delta_{share} \\ 0 & otherwise \end{cases}$$



Preserving diversity: Fitness sharing

- Fitness sharing: restrict the number of individuals within a niche by "sharing" their fitness

$$f'(i) = \frac{f(i)}{\sum_j sh(d(i,j))} \qquad sh(d) = \begin{cases} 1 - \left(\frac{d}{\delta_{share}}\right)^{\alpha} & if\ d \le \delta_{share} \\ 0 & otherwise \end{cases}$$

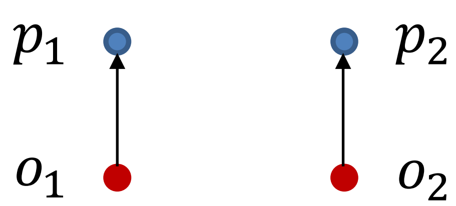


Preserving diversity: Crowding

- Crowding: the offspring only compete for survival with the similar parents

- For example,
  - The parent population is randomly paired
  - Each pair produces two offspring via recombination
  - These offspring are mutated and then evaluated
  - The distances between offspring and parents are calculated
  - Each offspring competes for survival with the similar parent

$$d(p_1, o_1) + d(p_2, o_2) < d(p_1, o_2) + d(p_2, o_1)$$

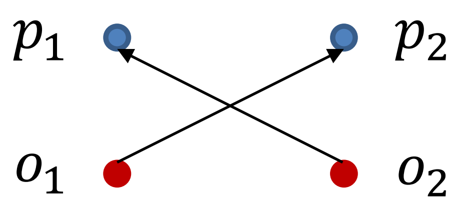


Preserving diversity: Crowding

- Crowding: the offspring only compete for survival with the similar parents

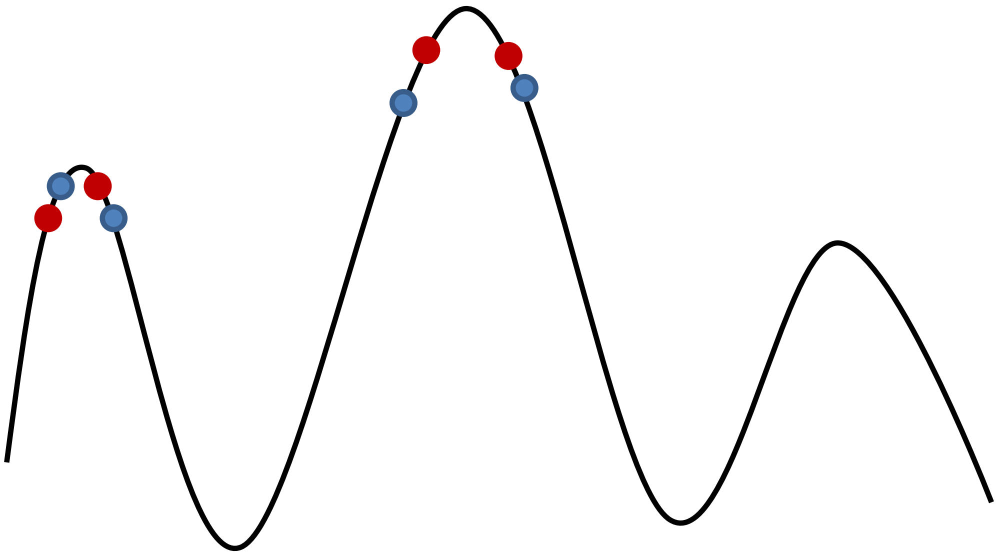
- For example,
  - The parent population is randomly paired
  - Each pair produces two offspring via recombination
  - These offspring are mutated and then evaluated
  - The distances between offspring and parents are calculated
  - Each offspring competes for survival with the similar parent

$$d(p_1, o_1) + d(p_2, o_2) > d(p_1, o_2) + d(p_2, o_1)$$



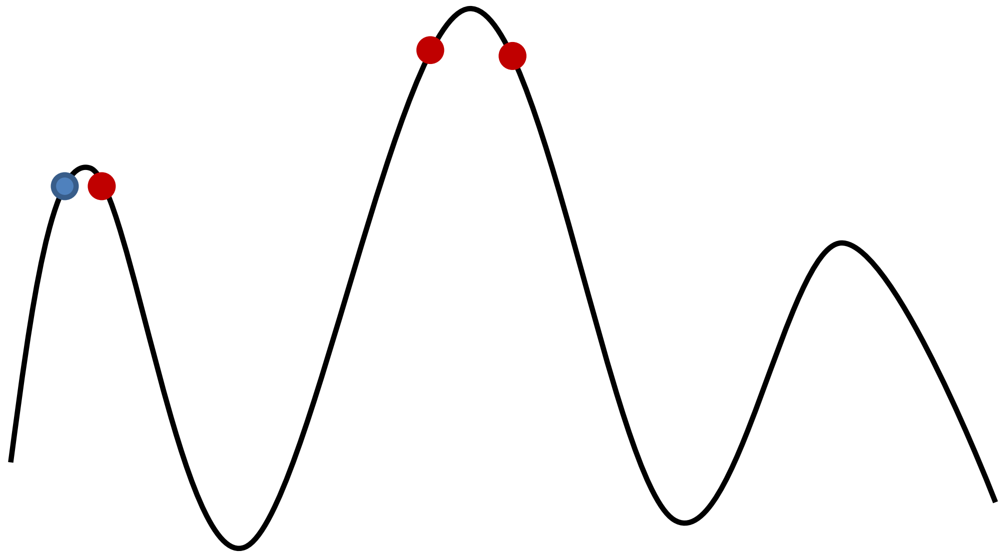
Preserving diversity: Crowding

- Crowding: the offspring only compete for survival with the similar parents



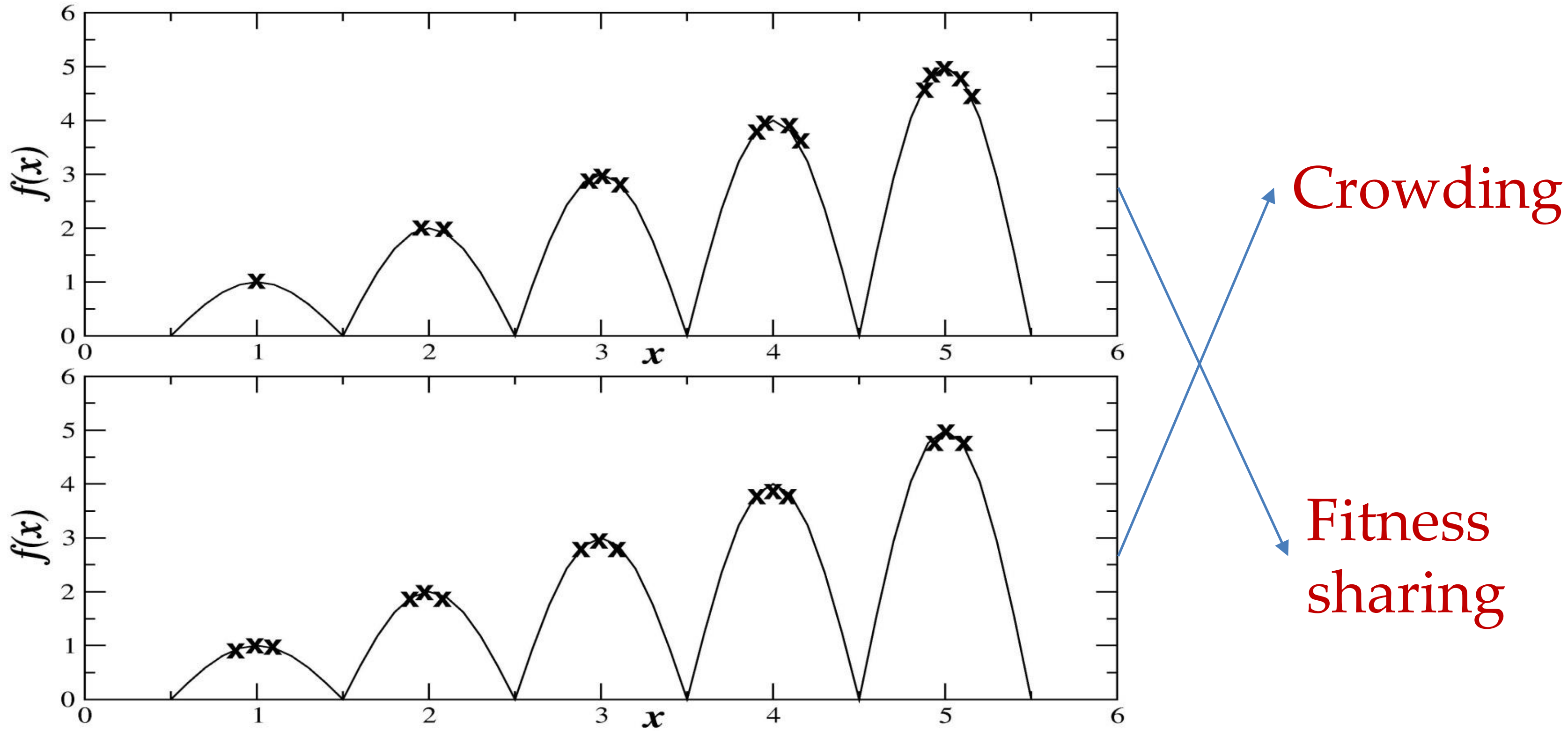
Preserving diversity: Crowding

- Crowding: the offspring only compete for survival with the similar parents



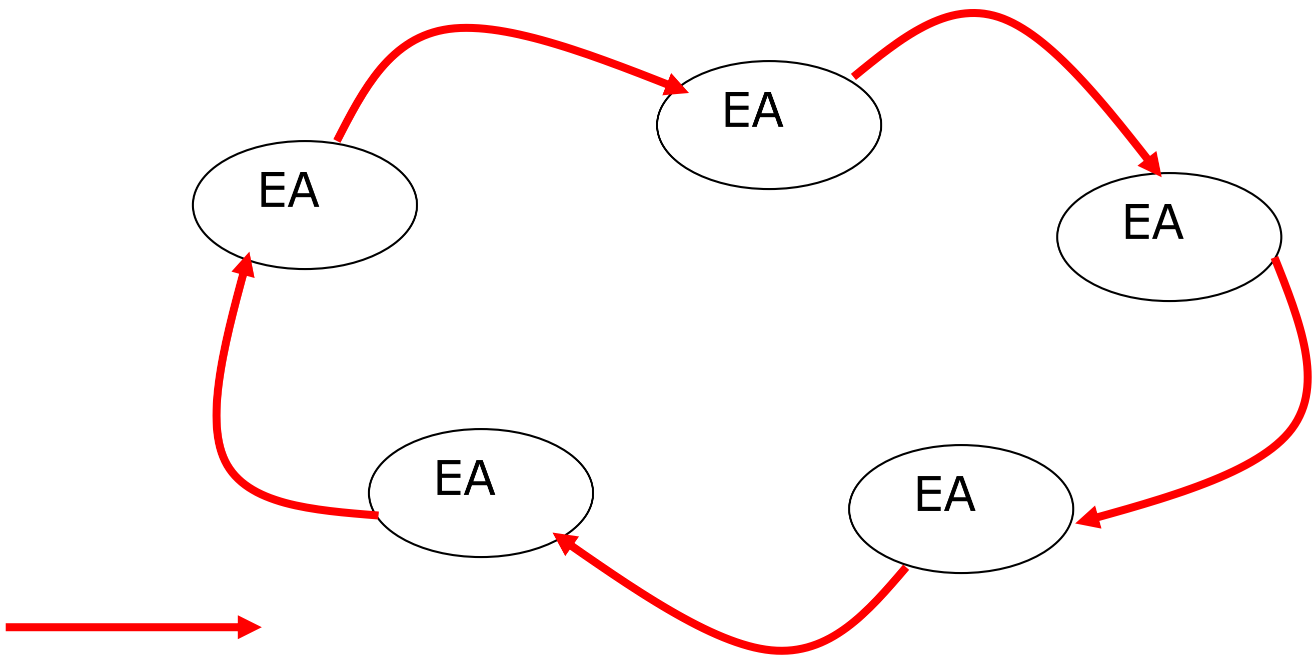
The population is equally distributed amongst niches

Preserving diversity: Fitness sharing and Crowding



Preserving diversity: Island model EAs

- Island model EAs: Run multiple sub-populations in parallel, and exchange individuals within neighbouring populations after a number of generations



Periodic migration of individuals between populations

Preserving diversity: Island model EAs

- How often to exchange individuals ?
  - if too quick, all sub-populations converge to the same solution
  - if too slow, time may be wasted
  - Suggested migration frequency: 25-150 generations
- How many, which individuals to exchange ?
  - usually 2-5, but depends on population size
  - Fitness-based selection or random selection
  - Copy (require survivor selection) or move (require symmetrical communication)
- How to divide the population into sub-populations ?
  - General rule: guarantee a minimum sub-population size and use more sub-populations

Operators can differ between the sub-populations

Summary

- Parent selection
- Survival selection
- Population diversity

References

- A. E. Eiben and J. E. Smith. Introduction to Evolutionary Computing. Chapter 5.

Assignment - 2

Preliminary

Task: apply evolutionary algorithms to solve submodular optimization problems

Deadline: Nov. 25

For reference: ACM GECCO Competition

https://gecco-2022.sigevo.org/Competitions#id_Evolutionary%20Submodular%20Optimisation

https://cs.adelaide.edu.au/~optlog/CompetitionESO2022.php