Sim-EA: An Evolutionary Algorithm Based on Problem Similarity

Krzysztof Michalak

Department of Information Technologies,
Institute of Business Informatics,
Wroclaw University of Economics, Wroclaw, Poland
krzysztof.michalak@ue.wroc.pl

Presentation Plan

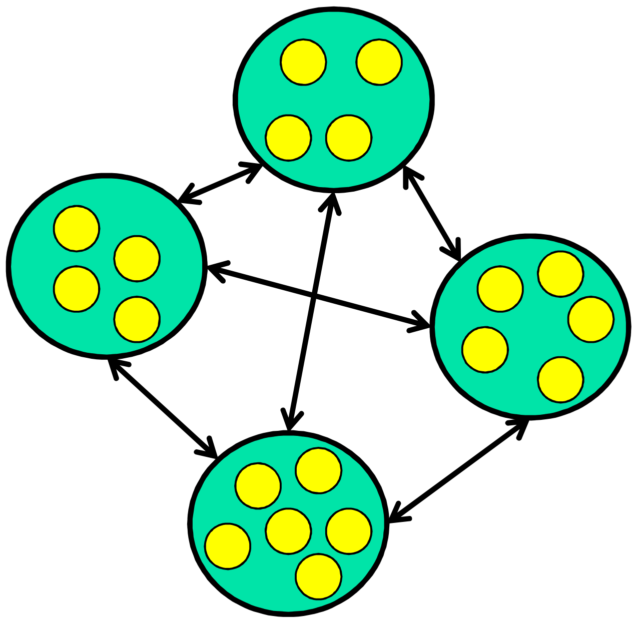
- Multipopulation EAs
- The Sim-EA algorithm
- Experiments and Results
- Conclusion

Multipopulation EAs

- Idea
  - instead of one big population use several smaller ones
  - each subpopulation can work in a different part of the search space

- Motivation
  - diversity preservation
  - multimodal problems
  - multiobjective optimization
  - dynamic optimization

- Island model
  - isolated environments (islands)
  - migration



The Sim-EA algorithm – overview

- A multipopulation algorithm…
- …but each subpopulation tackles a slightly different instance of the problem
- Island model + migration
- Migration influenced by similarities of problem instances
- Migrated specimens participate in genetic operations before selection phase
- Elitism (in each subpopulation separately)

The Sim-EA algorithm – migration

- Problem instance similarity matrix

$$S_{[N_{prob} \times N_{prob}]}$$

where:

N_{prob} – the number of problem instances

higher values = more similar instances

- Migration stategies
  - 1-nearest-*n*-best – migrate *n* best specimens from the most similar subproblem
  - 1-uniform-*n*-best – migrate *n* best specimens from a subproblem chosen at random
  - none – no migration

The Sim-EA algorithm – migration

- Incoming specimens are added one by one
  - x – incoming specimen
  - w – the weakest specimen in target population
  - binary tournament between x and w
  - if x wins it replaces w

The Sim-EA algorithm

Algorithm 1. The overview of the Sim-EA algorithm

IN:
- N_{gen} - the number of generations
- N_{pop} - the size of each subpopulation
- N_{prob} - the number of problem instances
- N_{imig} - the number of migrated specimens

Calculate the problem instance similarity matrix $S_{[N_{prob} \times N_{prob}]}$

Initialize subpopulations $P_1, P_2, \ldots, P_{N_{prob}}$.

```
for g = 1,...,N_gen do
    Apply genetic operators
    for d = 1,...,N_prob do                          } select immigrants
        s = argmax_t(S_{d,t})
        P'_d = the N_imig best specimens from P_s
    end for
    for d = 1,...,N_prob do                          } perform migration
        for x ∈ P'_d do
            P'_d = P'_d − {x}
            w = the weakest specimen in P_d
            P_d = P_d − {w}
            b = BinaryTournament(w, x)
            P_d = P_d ∪ {b}
        end for
    end for
    Apply genetic operators
    for d = 1,...,N_prob do                          } selection with elitism
        e = the best specimen in P_d
        P_d = Select(P_d\{e}, N_pop − 1)
        P_d = P_d ∪ {e}
    end for
end for
```

Proceedings, page 193

Test Problem

- The Travelling Salesman Problem (TSP) with $K = 50$ cities
  - find the shortest route through K cities, visiting each city only once
  - distances given in a cost matrix $C^{(1)}_{[K \times K]}$
  - each solution is a permutation
  - minimization problem:

$$\text{minimize } f(p) = c_{p(K)p(1)} + \sum_{i=1}^{K-1} c_{p(i)p(i+1)}$$
$$\text{subject to } p \in \mathrm{P}_K,$$

where:

P_K - the set of all permutations of numbers 1, …, K

Test Problem

- N_{prob} = 20 cost matrices

$$C^{(1)}_{[K\times K]},\ldots,C^{(20)}_{[K\times K]}$$

- In $C^{(1)}_{[K\times K]}$ the elements were drawn from U[0, 100]

- $C^{(j)}_{[K\times K]}$ was generated from $C^{(j-1)}_{[K\times K]}$ by replacing 1 / N_{prob} (1/20 = 5%) of elements by random values

- Similarity calculated as:

$$S_{i,j} = -\sum_{p=1}^{K}\sum_{q=1}^{K}(C^{(i)}_{p,q} - C^{(j)}_{p,q})^2$$

Experiments

- 30 iterations for each migration strategy
- Algorithm parameters:

Parameter name	Value
Number of subproblems (N_{prob})	20
Problem size (the number of cities, K)	50
Number of generations (N_{gen})	200
Population size (N_{pop})	100
Random inverse rate for the inver-over operator (δ_i)	0.02

Results

- Median values of the travel cost obtained by each of the migration strategies for each of the subproblems

Subproblem	1-nearest N-best	1-uniform N-best	none	Subproblem	1-nearest N-best	1-uniform N-best	none
1	**171.3077**	176.1562	176.0278	11	**219.3591**	221.2456	221.6789
2	**179.3029**	182.5553	183.6177	12	**223.8487**	226.7204	227.2988
3	**190.1753**	193.9355	195.0356	13	211.1367	**210.3828**	214.3309
4	**194.6214**	198.8933	201.1124	14	**212.1491**	214.721	214.4843
5	**186.2072**	192.4362	192.5718	15	**222.9707**	223.0239	224.8621
6	**180.9736**	186.2206	187.4909	16	**228.5528**	230.5679	229.6890
7	**188.1285**	191.1580	192.2368	17	**263.6930**	266.6074	268.6715
8	**193.7582**	194.4849	198.1625	18	**262.3891**	263.0081	266.0693
9	**202.9813**	206.9935	209.6731	19	**257.2304**	258.7912	260.6183
10	**223.4921**	225.8248	227.4927	20	**249.1812**	252.2827	253.0103

Results

- The p-values for the null hypothesis that the 1-nearest-N-best strategy gives worse results than each of the two remaining strategies

Sub-problem	vs. none		vs. 1-uniform-N-best		Sub-problem	vs. none		vs. 1-uniform-N-best	
	p-value	interp.	p-value	interp.		p-value	interp.	p-value	interp.
1	0.0001891	signif.	0.00017423	signif.	11	0.0082167	signif.	0.11093	insignif.
2	0.00061564	signif.	0.0054597	signif.	12	0.00096266	signif.	0.0014839	signif.
3	0.00052872	signif.	0.0082167	signif.	13	0.057096	insignif.	0.97539	worse
4	3.1123e-005	signif.	2.163e-005	signif.	14	0.013194	signif.	0.17138	insignif.
5	0.0001057	signif.	0.0003065	signif.	15	0.020671	signif.	0.89364	insignif.
6	6.3391e-006	signif.	8.4661e-006	signif.	16	0.40483	insignif.	0.13059	insignif.
7	1.2381e-005	signif.	0.0064242	signif.	17	0.00048969	signif.	0.031603	signif.
8	0.00066392	signif.	0.036826	signif.	18	0.0019646	signif.	0.4908	insignif.
9	9.3157e-006	signif.	0.00035888	signif.	19	0.0024147	signif.	0.17791	insignif.
10	0.0017088	signif.	0.0046818	signif.	20	0.00083071	signif.	0.0046818	signif.

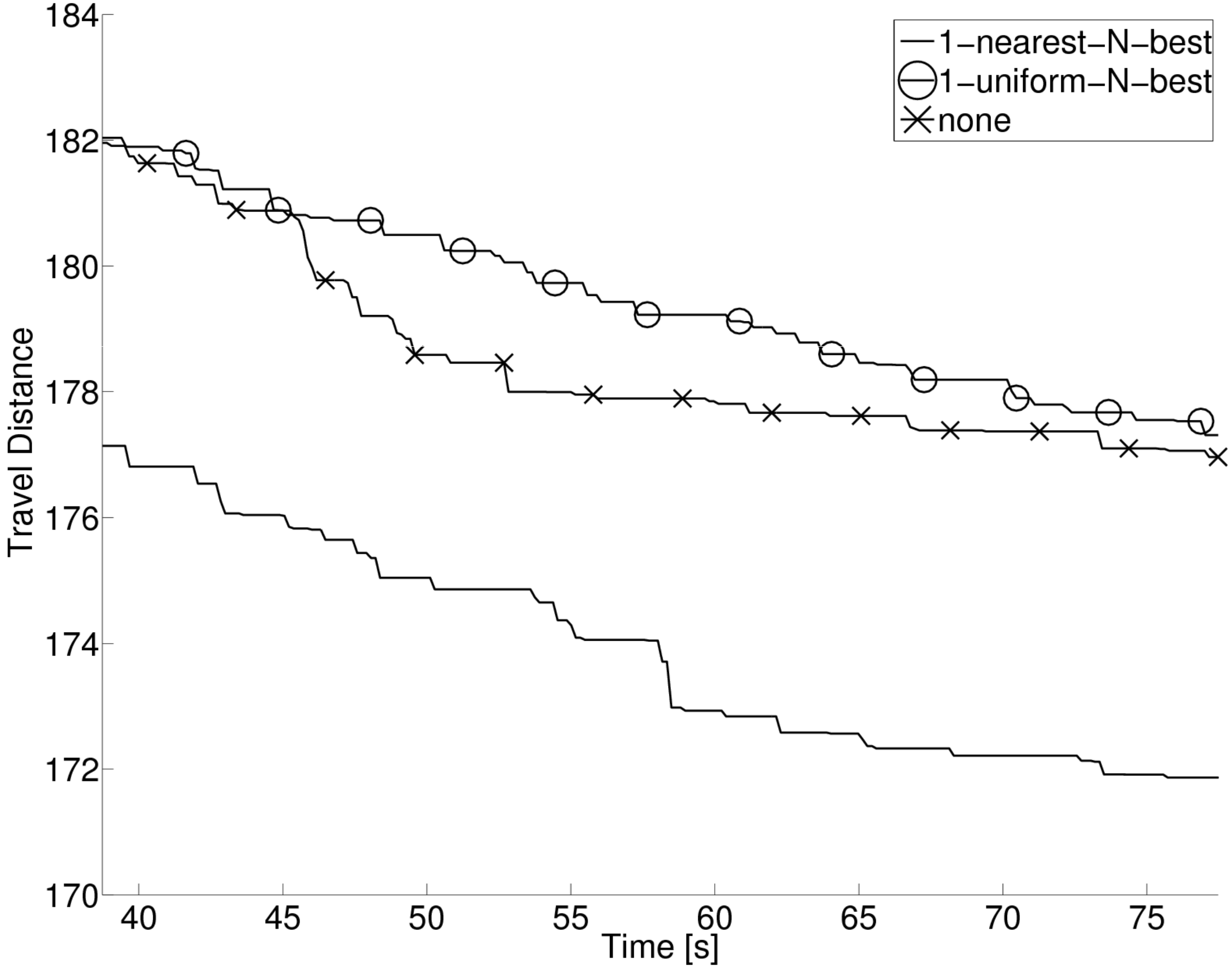
- Wilcoxon rank test, which does not assume normality was used for statistical testing

Graphs

- Median values of the best specimen travel distance calculated over all 30 runs
- Travel distance plotted against total calculation time
- The total calculation time includes the time used for calculating the elements of the similarity matrix

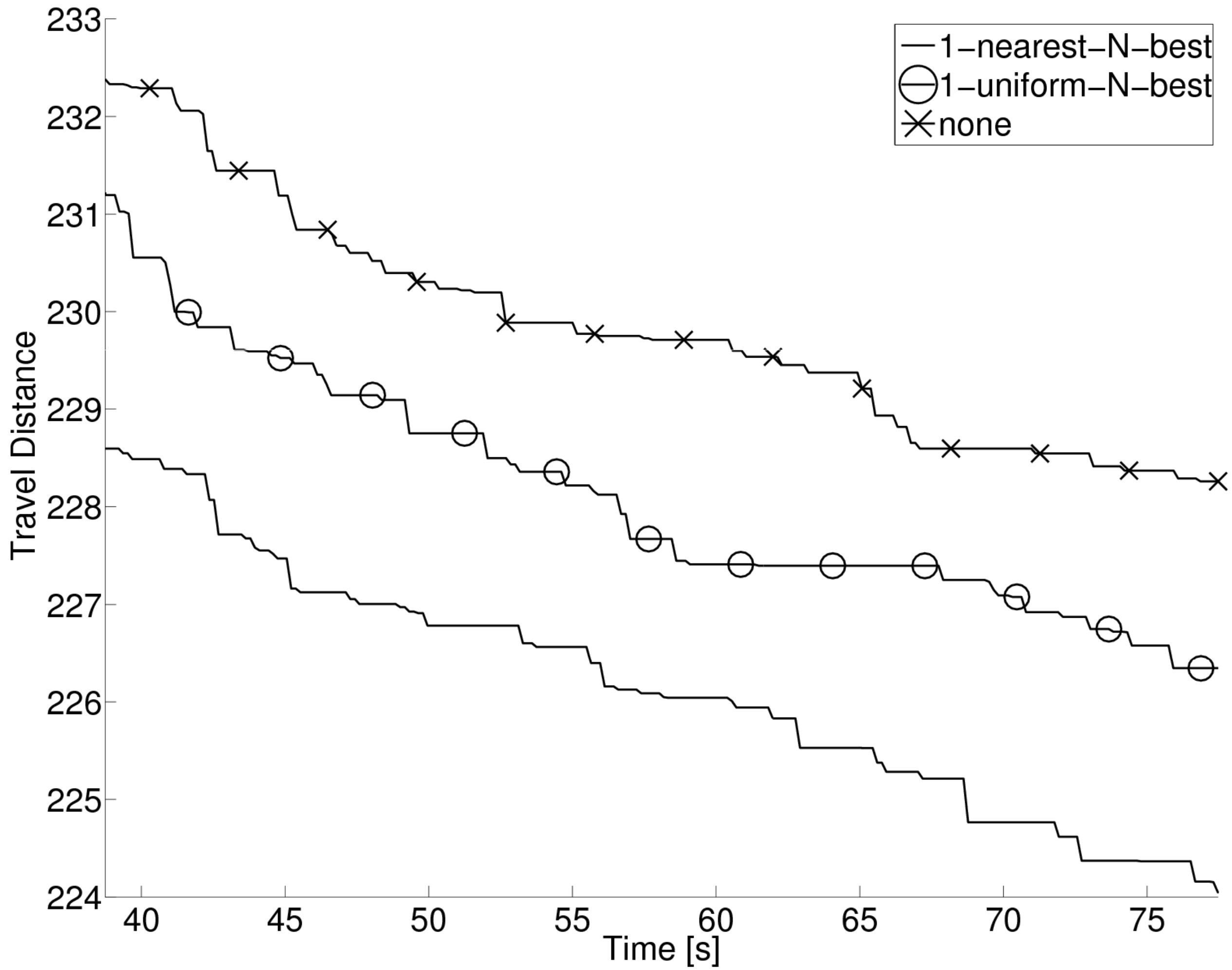
Graphs

- Subproblem #1



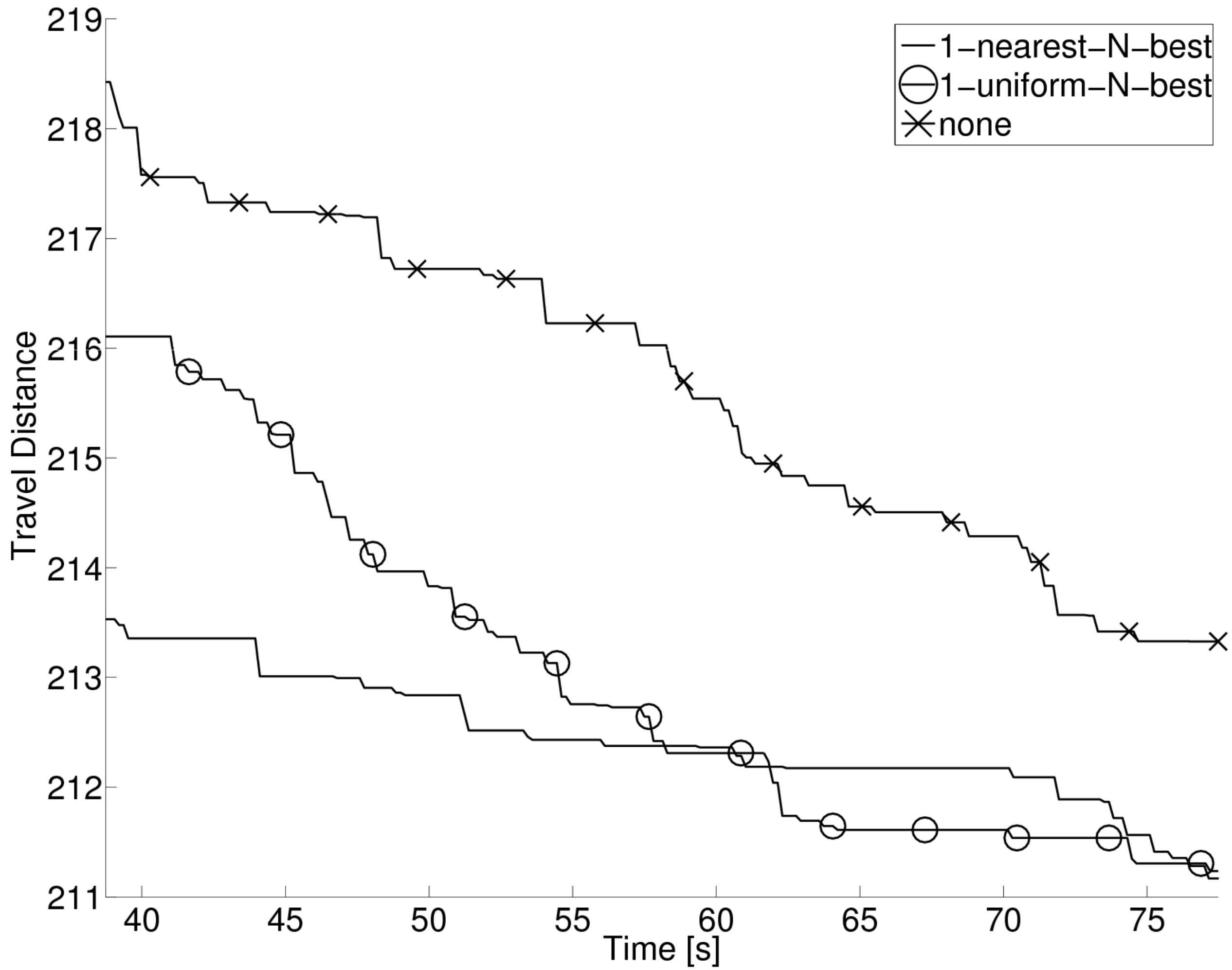
Graphs

- Subproblem #10



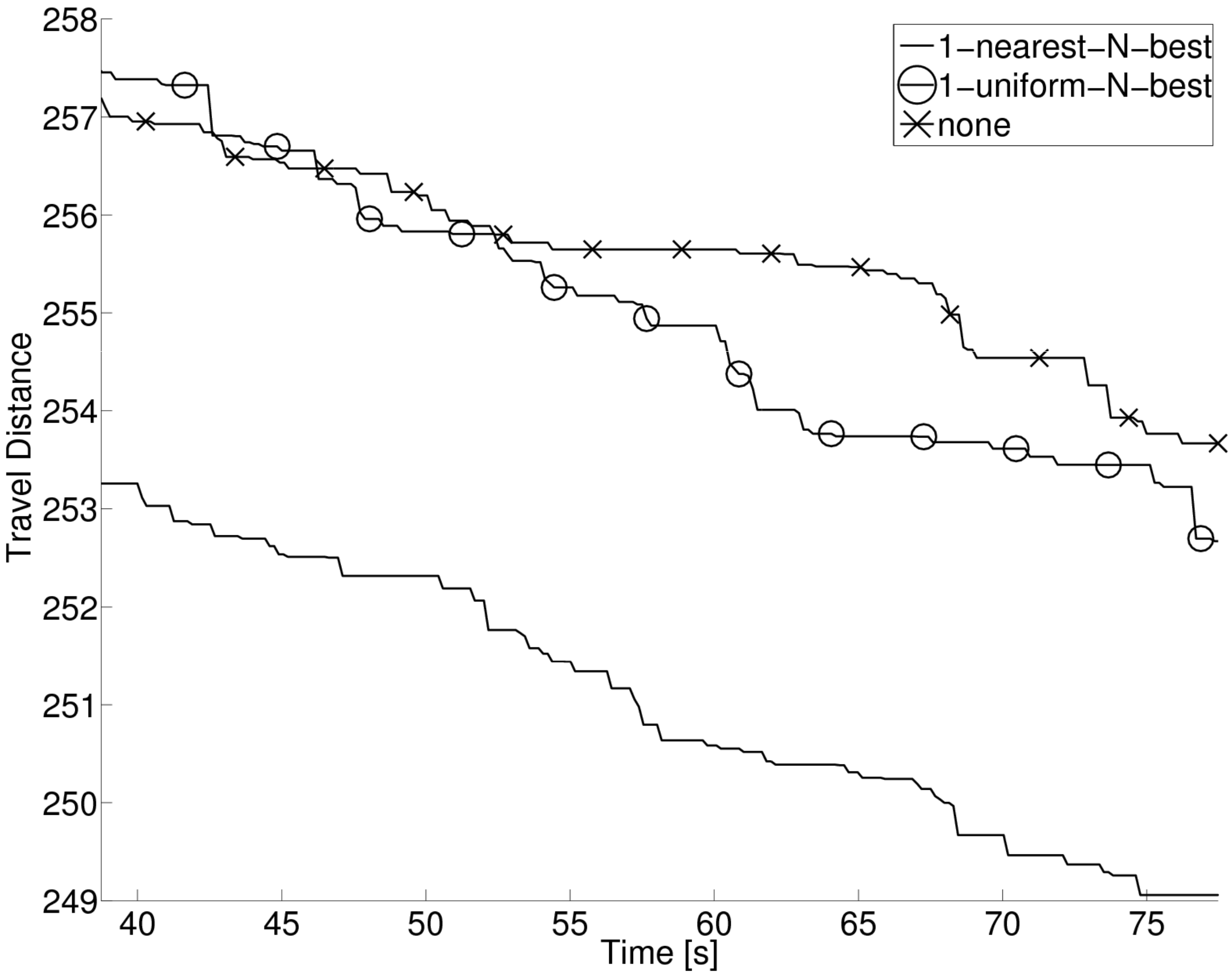
Graphs

- Subproblem #13



Graphs

- Subproblem #20



Conclusion

- An algorithm was proposed which utilizes a multipopulation approach to solve several similar instances of an optimization problem simultaneously
- Tests performed on 20 Travelling Salesman Problem instances
- Varying degree of similarity between the instances
- The 1-nearest-N-best migration strategy worked better than the 1-uniform-N-best strategy
- Without migration the results are even further deteriorated
- Further work: updating the similarity measure during the algorithm runtime based on information discovered during the optimization process

Thank you!

The inver-over operator

```
select (randomly) a city c from S'
repeat
{
  if (rand() ≤ p)
    select the city c' from the remaining cities in S'
  else
  {
    select (randomly) an individual from P
    assign to c' the 'next' city to the city c in the selected individual
  }
  if (the next city or the previous city of city c in S' is c')
    exit from repeat loop
  inverse the section from the next city of city c to the city c' in S'
  c = c'
}
```

Source: Tao, G., Michalewicz, Z. „Inver-over Operator for the TSP", Parallel Problem Solving from Nature - PPSN V, Lecture Notes in Computer Science Volume 1498, 1998, pp 803-812