14: Working with EAs

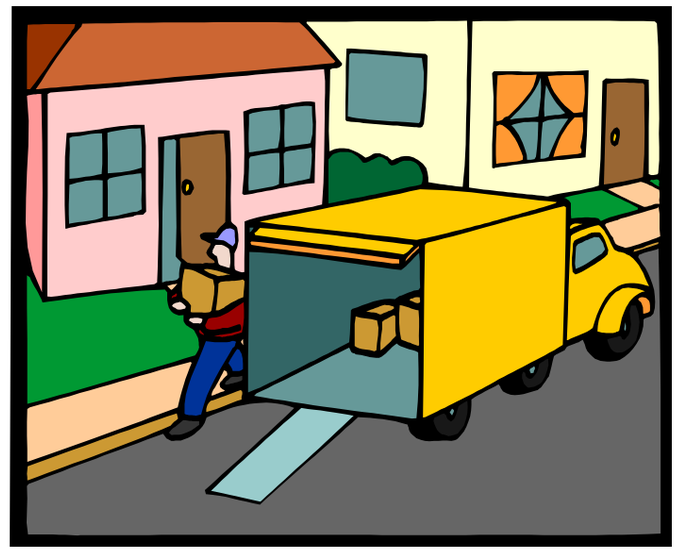
- Goals of problem-solving
- Algorithm design
- Rules of experiments
- Performance measures
- Textbook Chapter 9

What do you want an EA to do?

- Design (one-off) problems
- Repetitive (on-line) problems

- Optimize national road network

- Optimize package delivery route



Perspectives of goals

- Design problems:
  - find a very good solution at least once
- Repetitive problems:
  - find a good solution at almost every run
- Scientific research:
  - must meet scientific standards

Academic experimentation goals

- Show that EC is applicable in a problem domain
- Show that an EA with new features is better than benchmark
- Find the best setup for parameters of a given TA
- Obtain insights into algorithm behavior
- Obtain insights into the problems
- See how the EA scales up with problem size
- See how the performance is influenced by parameters of the problems and the algorithm

Algorithm design

- Design a representation
- Design a way of mapping a genotype to a phenotype
- Design a way of evaluating an individual
- Design suitable mutation operator(s)
- Design suitable recombination operator(s)
- Decide how to select individuals to be parents
- Decide how to select individuals for the next generation
- Decide how to start
- Decide how to stop

Basic rules of experimentation

- EAs are stochastic -> never draw any conclusion from a single run
  - perform sufficient number of independent runs
  - use statistical measures (mean, SD, CI)
  - use statistical tests to assess reliability of conclusions
- EA experimentation is about comparison -> always do a fair competition
  - use the same amount of resources for the competitors
  - try different computational limits
  - use the same performance measures

Things to measure

- Average result in a given time
- Average time for given result
- Proportion of runs within % of target
- Best result over n runs
- Amount of computing required to reach target in giving time with % confidence
- ……..

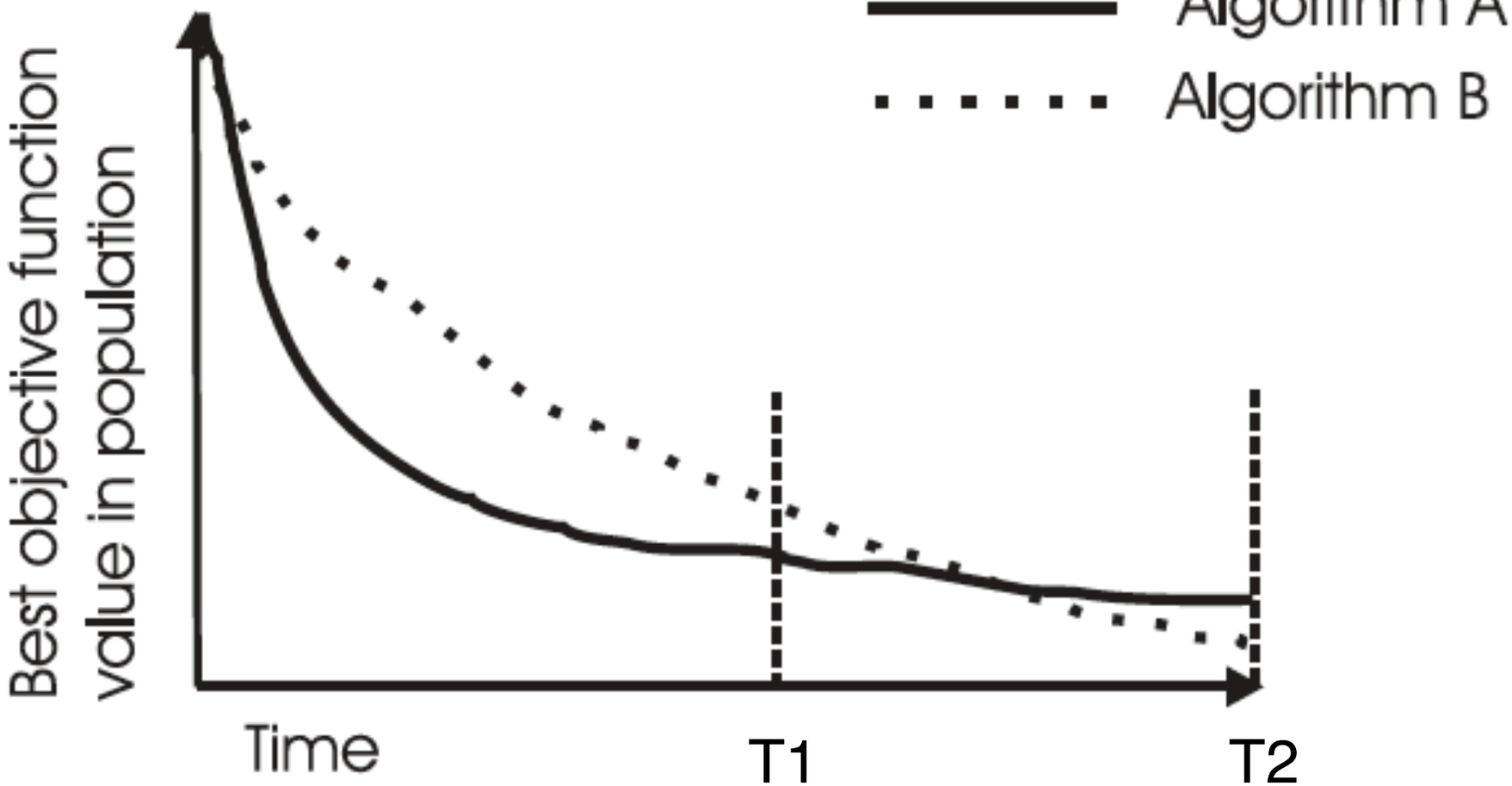
Measures

- Performance measures (off-line)
  - Efficiency
  - Effectiveness
- Working measures (on-line)
  - Population distribution (genotypic)
  - Fitness distribution
  - Improvement per time unit or per genetic operator

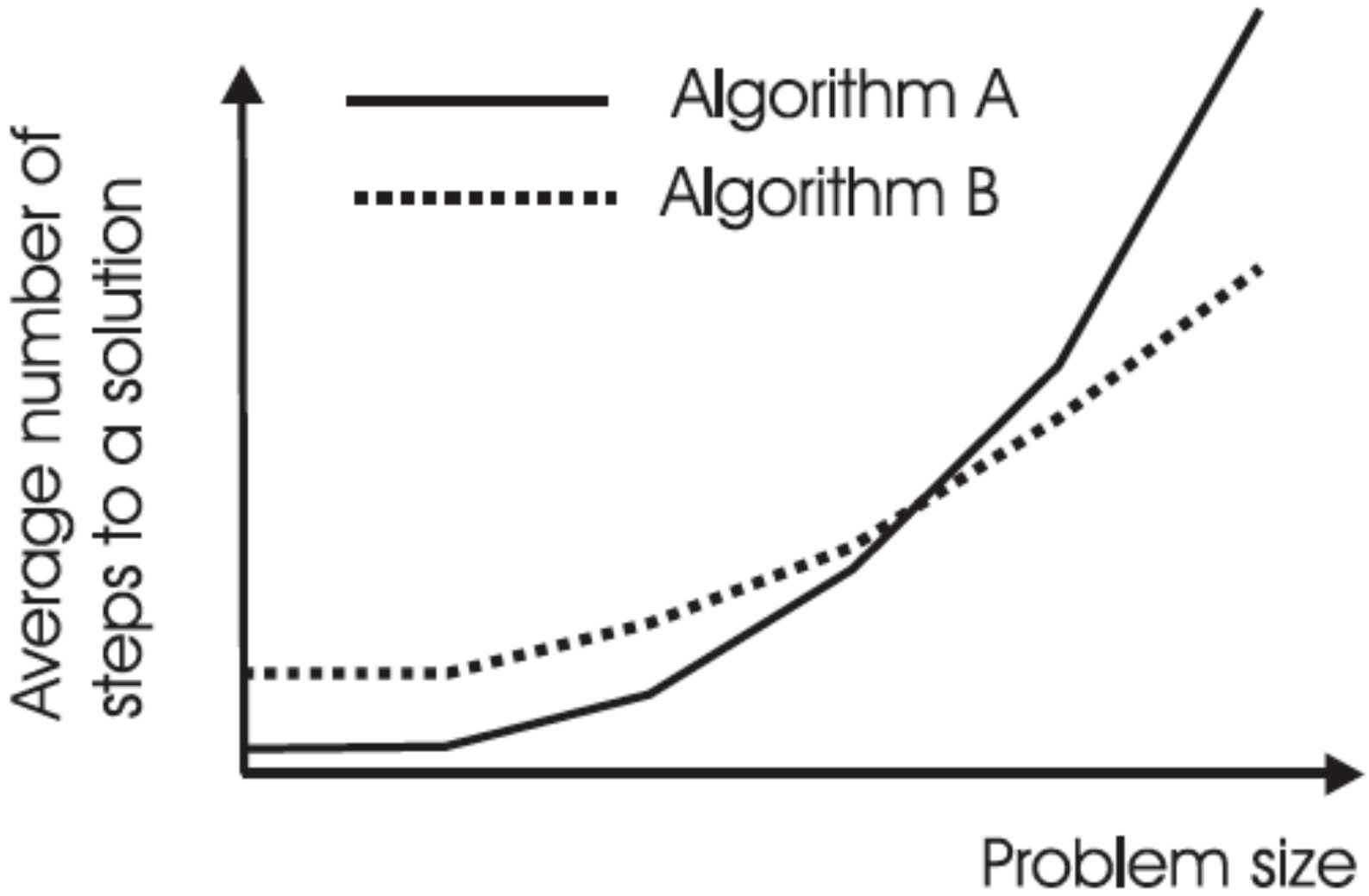
Performance measures

- Number of generated points in the search space (fitness evaluations)
- AES: average number of evaluations to a solution
- SR: success rate (% of runs finding a solution with acceptable quality)
- MBF: mean best fitness at termination (best per run, mean over runs)
- SR vs. MBF
  - SR might be difficult to define for some problems
  - MBF is always a valid measure
  - low SR and high MBF
  - high SR and low MBF

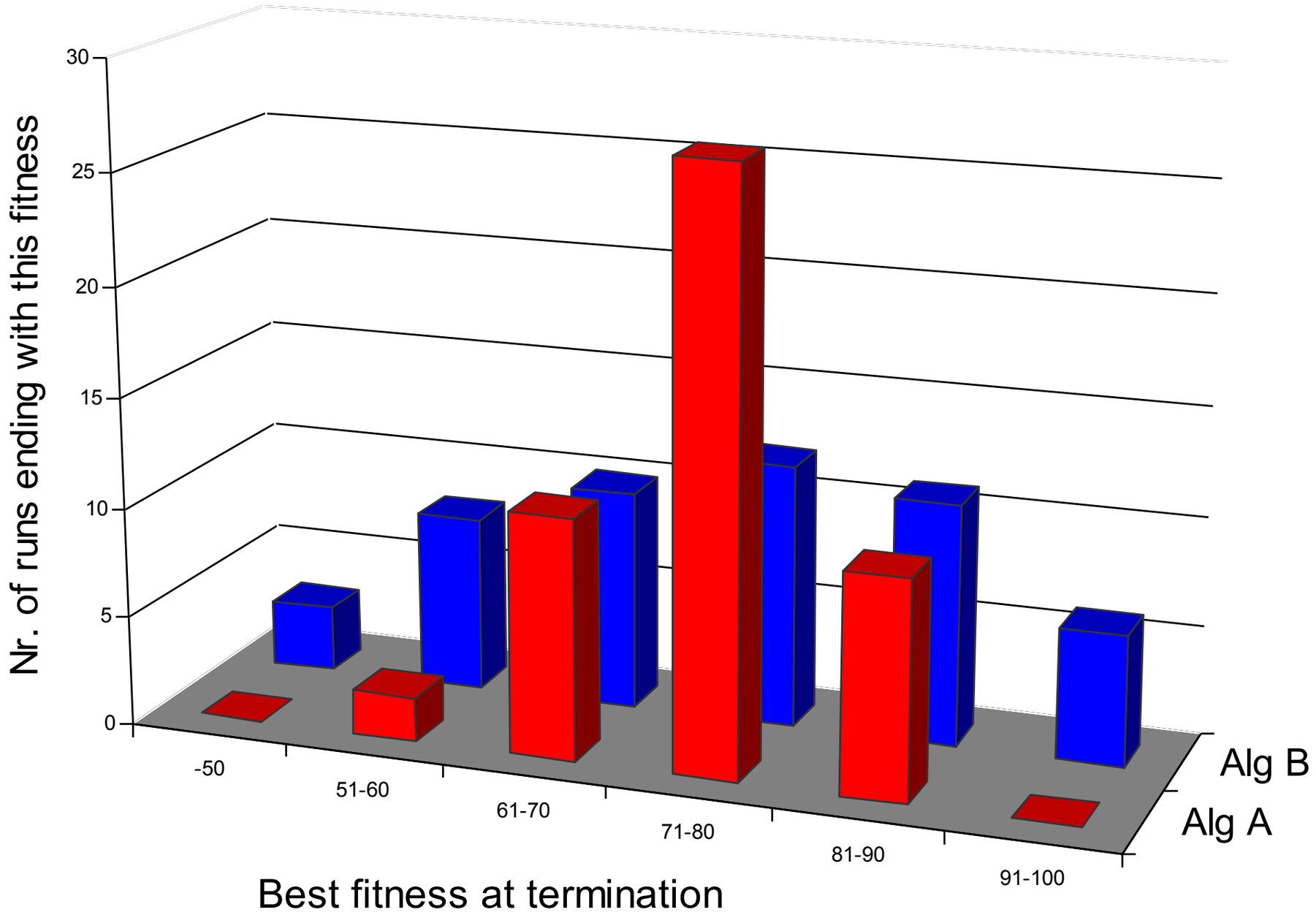
Example I


Algorithm A
Algorithm B
Best objective function value in population
Time
T1
T2

Example2


Algorithm A
Algorithm B
Average number of steps to a solution
Problem size

Peak vs. average performance



Statistical comparison and significance

- Algorithms are stochastic, results have element of "luck"
- If a claim is made "mutation A is better than mutation B", need to show statistical significance of comparisons
- Fundamental problem: two series of samples from the same distribution may have different averages and SDs
- Tests can show if the differences are significant or not

Example

Trial	Old Method	New Method
1	500	657
2	600	543
3	556	654
4	573	565
5	420	654
6	590	712
7	700	456
8	472	564
9	534	675
10	512	643
Average	545.7	612.3

Example

Trial	Old Method	New Method
1	500	657
2	600	543
3	556	654
4	573	565
5	420	654
6	590	712
7	700	456
8	472	564
9	534	675
10	512	643
Average	545.7	612.3
SD	73.5962635	73.5473317
T-test	**0.07080798**	

Statistical tests

- Student *t*-test:
  - data taken from continuous interval or close approximation
  - normal distribution
  - similar variances for too few data points
  - similar sized groups of data points
- Other tests:
  - Wilcoxon: preferred to *t*-test where distribution is not known
  - F-test: see if two samples have different variances

Good example: problem setting

- I invented myEA for problem X
- Looked and found three other EAs and a traditional benchmark heuristic for problem X in the literature
- Asked myself when and why myEA is better

Good example: experiments

- Found/made problem instance generator for problem X with two parameters (*n* - problem size; *k* - some problem specific indicator)
- Select five values for *k* and five values for *n*
- Generated 100 problem instances for all combinations
- Executed all algorithms on each instance 100 times
- Recorded AES, SR, and MBF with same computational limit

Good example: evaluation

- Arrange results in 3D: (*n*,*k*) + performance with special attention to the effect of *n* as for scale-up
- Assessed statistical significance of results
- Found the niche for myEA (weak in some cases and strong in some cases, comparable otherwise)
- Analyzed the specific features and the niches of each algorithm thus answering the "why question"
- Learned a lot about problem X and its solvers
- Achieved generalizable results, or at least claims with well-identified scope based on solid data
- Facilitated reproducing my results -> further research

Some tips

- Be organized
- Decide what you want and define appropriate measures
- Choose test problems carefully
- Make an experiment plan
- Perform sufficient number of runs
- Keep all experimental data
- Use good statistics
- Present results well
- Watch the scope of your claims
- Aim at generalized results
- Publish code and data (reproducibility and open science)