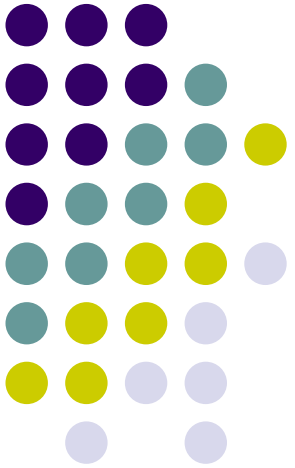
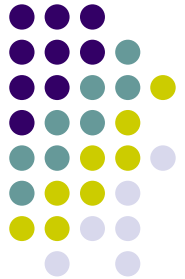
Problem Solving and Search in Artificial Intelligence

Learning and Search



Nysret Musliu
Database and Artificial Intelligence Group,
Institut für Informationssysteme, TU-Wien

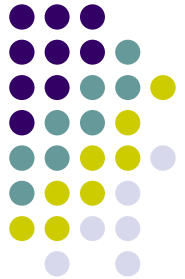
Motivation



- Metaheuristic techniques usually include several parameters
  - Tabu search: length of tabu list, type of memory, …
  - Simulated annealing: start and end temperature, decrease of temperature…
  - Iterated local search: size of perturbation, acceptance criteria, running time of local search procedure …
  - Evolutionary algorithms: population size, crossover rate, mutation rate,…
  - …
- Different components can be used
  - Neighborhood structure
  - Mutation type/crossover type
  - …

Finding appropriate parameters/components to be used is crucial for the performance of heuristics

Motivation (Algorithm selection)



- Usually several search algorithms are available for solving a particular problem
- No free lunch theorem

"…for any algorithm, any elevated performance over one class of problems is offset by performance over another class" [1]

"any two algorithms are equivalent when their performance is averaged across all possible problems" [2]

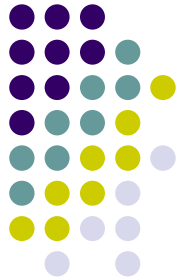
How to select the best algorithm for a specific instance?

[1] David Wolpert, William G. Macready: No free lunch theorems for optimization. IEEE Transac. Evolutionary Computation 1(1): 67-82 (1997)
[2] Wolpert, D.H., and Macready, W.G. (2005) "Coevolutionary free lunches," IEEE Transac. on Evolutionary Computation, 9(6): 721-735

Algorithm configuration (setting of parameters)

- Parameters are determined manually
- Automated algorithm configuration
  - Off-line parameter setting
  - On-line parameter setting

Manual Algorithm Configuration (I)



- Select different configuration for parameters
- Select representative instances for the problem to be solved
- Run experiments on instances with each parameter configuration (typically many runs per instance should be executed)
- Select the best configuration based on quality of solutions, running time of algorithm, …
- Statistical Analysis

Manual Algorithm Configuration (II)

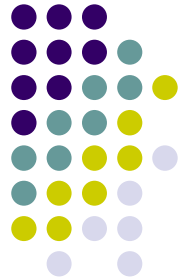
- Disadvantages of manual configuration
  - Time consuming for the designer of algorithms
  - Limited number of configuration can be tested
  - Hard to find the best configuration

  - Alternatives:
    - Automated algorithm configuration

Automated Algorithm Configuration

- Input for off-line configuration problem:
  - Algorithm A
  - A set of parameter configurations
  - A set of input instances
- Problem:
  - Find parameter configuration that gives the best results on the input instances (e.g. solutions with best quality, time performance, …)
- This problem is a search problem - > Number of solutions is equal to number of possible parameter configuration

Off-line algorithm configuration: Example I

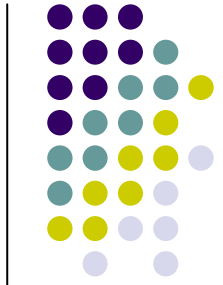


ParamILS [3]:

- Applies iterated local search to find the best configuration
  - Search space: all possible parameter configurations
  - Objective function: the performance of the algorithm with a specific configuration
  - Neighborhood: modification of one parameter value at a time
  - Additional mechanism to speed-up the algorithm (by avoiding unnecessary runs)
- Applied for configuration of CPLEX, SAT algorithms, …

[3] Frank Hutter, Holger H. Hoos, Kevin Leyton-Brown, Thomas Stützle: ParamILS: An Automatic Algorithm Configuration Framework. J. Artif. Intell. Res. (JAIR) 36: 267-306 (2009) (http://www.cs.ubc.ca/labs/beta/Projects/ParamILS/)

Off-line configuration: Example II



GGA - A Gender-Based Genetic Algorithm [4]

- Introduces a gender separation
- Speedup with parallelization

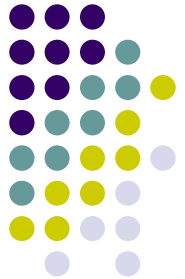
ISAC - Instance-Specific Algorithm Configuration [5]

- Integrates GGA and stochastic offline programming
- Training instances are clustered based on some features
- Best parameters are found for each cluster with GGA
- Offers selection of best parameters based on features of an input instance
- Applied for Set Cover, SAT and Mixed Integer Programming

[4] Carlos Ansótegui, Meinolf Sellmann, Kevin Tierney: A Gender-Based Genetic Algorithm for the Automatic Configuration of Algorithms. CP 2009: 142-157

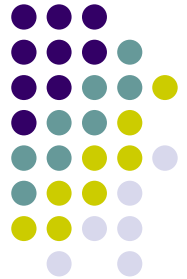
[5] Serdar Kadioglu, Yuri Malitsky, Meinolf Sellmann, Kevin Tierney: ISAC - Instance-Specific Algorithm Configuration. ECAI 2010: 751-756

On-line parameter setting



- Parameters change based on feedback during the search
- A parameter can change based on simple rules:
  - Increase tabu length or mutation size if diversification is needed
  - Apply neighborhood relations that imporved most of the times solutions
  - …
- More sophisticated techniques use machine learning techniques (for example reinforcement learning)

Adaptive techniques: Example I

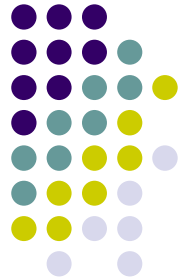


Reactive tabu search [6]

- The prohibition T is determined based on feedback during the search
- T=1 in the start of the search
- T increases for 1 if diversification is needed
- The evidence that diversification is needed appears if for example
  - Previous solutions are repeated
  - The solutions have a short distance to the previous solutions
- T decreases if diversification is not needed

[6] Roberto Battiti, Giampietro Tecchiolli: The Reactive Tabu Search. INFORMS Journal on Computing 6(2): 126-140 (1994)

Adaptive techniques: Example II



Application of Reinforcement Learning (RL)

Online Control of Evolutionary Algorithms [7]

- A reinforcement method runs simultaneously with the evolutionary algorithm
- EA parameters: population size, tournament proportion, mutation probability, crossover probability
- RL learning changes above during the search based on the progress (best fitness, mean fitness, standard deviation, …) of EA between two time points

[7] A. E. Eiben, Mark Horvath, Wojtek Kowalczyk, Martijn C. Schut: Reinforcement Learning for Online Control of Evolutionary Algorithms. ESOA 2006: 151-160

Algorithm selection

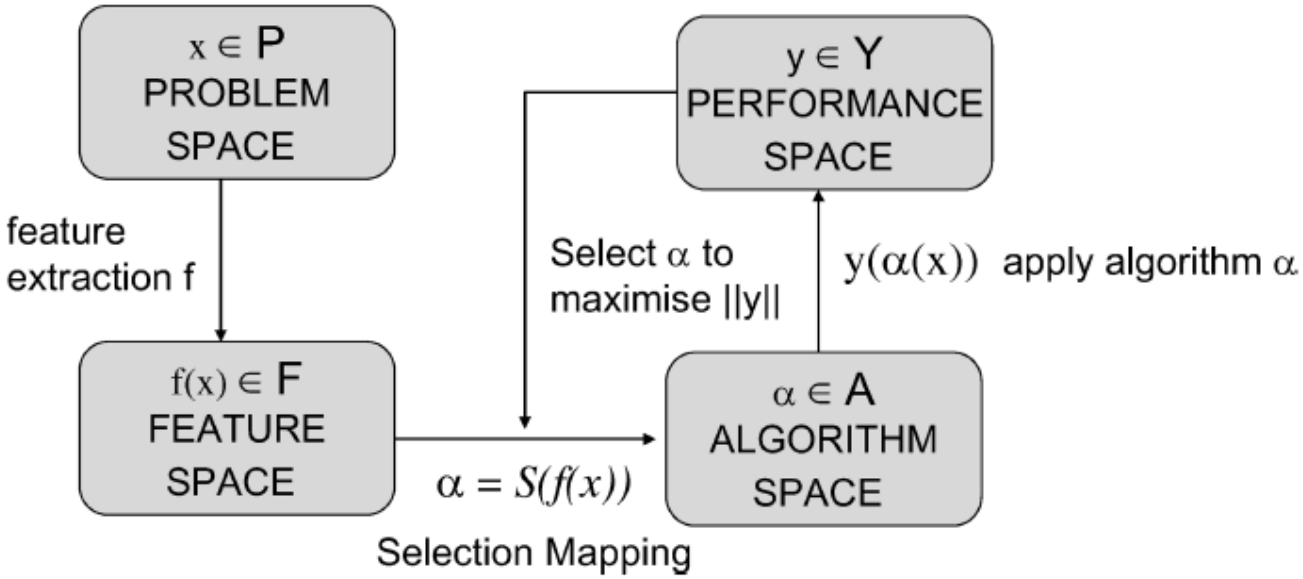
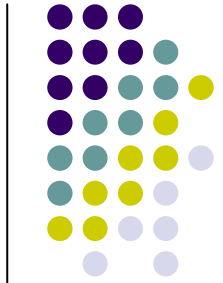


Figure taken from [9]

[8] John R. Rice: The Algorithm Selection Problem. Advances in Computers 15: 65-118 (1976)

[9] Kate Smith-Miles: Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Comput. Surv. 41(1): (2008)

Algorithm selection



Input (see [8] and [9]):

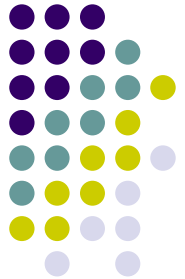
- Problem space P that represents the set of instances of a problem class
- A feature space F that contains measurable characteristics of the instances generated by a computational feature extraction process applied to P
- Set A of all considered algorithms for tackling the problem
- The performance space Y represents the mapping of each algorithm to a set of performance metrics

Problem:

For a given problem instance *x E P, with features f(x) E F, find the selection mapping S(f(x)) into algorithm space , such that the selected algorithm a E A maximizes the performance mapping y(a(x)) E Y*

[8] John R. Rice: The Algorithm Selection Problem. Advances in Computers 15: 65-118 (1976)
[9] Kate Smith-Miles: Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Comput. Surv. 41(1): (2008)

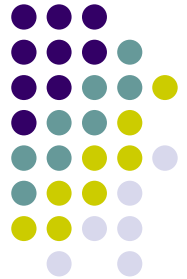
Algorithm selection



- An important issue is the selection of appropriate features
  - Example: Selection of sorting algorithms based on features ([10]):
    - Degree of pre-sortedness of the starting sequence
    - Length of sequence
- A supervised machine learning approach can be used to select the algorithm to be used based on features of the input instance
- A training set with instances (and their features) and best performing algorithm should be provided to the supervised machine learning algorithms to train them

[9] Kate Smith-Miles: Cross-disciplinary perspectives on meta-learning for algorithm selection. ACM Comput. Surv. 41(1): (2008)
[10] Guo, H. 2003. Algorithm selection for sorting and probabilistic inference: A machine learning-based approach.
Ph.D. dissertation, Kansas State University.

Algorithm selection for sorting [10]



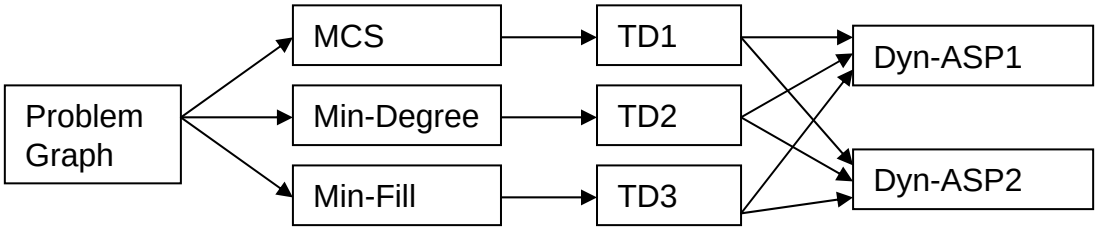
- P=43195 instances of random sequences of different sizes and complexities
- A=5 sorting algorithms (InsertionSort, ShellSort, heapSort, mergeSort, QuickSort)
- Y=algorithm rank based on CPU time to achieve sorted sequence
- F=3 measures of presortedness and length of sequences (size)
- Machine learning methods: C4.5, Naïve Bayes, Bayesian network learner

Different other examples are given in [9]

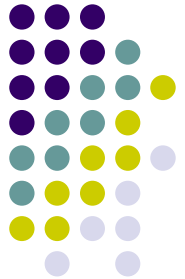
[10] Guo, H. 2003. Algorithm selection for sorting and probabilistic inference: A machine learning-based approach. Ph.D. dissertation, Kansas State University.

Algorithm selection examples

Select one of algorithms based on tree decomposition features (tree width, size of tree, …)



Other approaches



- Hyperheuristics [11]
  - Used to select between different low level heuristics
  - Ongoing competition in Nottingham
    http://www.asap.cs.nott.ac.uk/chesc2011/
- Dynamic Algorithm selection with reinforcement learning [12]

[11] Burke, E. K., M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and R. Qu (2010). Hyper-heuristics: A Survey of the State of the Art, School of Computer Science and Information Technology, University of Nottingham, Computer Science Technical Report No. NOTTCS-TR-SUB-0906241418-2747.

[12] Michail G. Lagoudakis, Michael L. Littman: Algorithm Selection using Reinforcement Learning. ICML 2000: 511-518