CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Cybernetics

1. Revision of Prerequisites. Hybrid Algorithms. Memetic Algorithms.

Petr Pošík

This lecture is based on the following works:
'A taxonomy of hybrid metaheuristics'
by E. G. Talbi, Journal of Heuristics, 2002,
and
'A tutorial for competent memetic algorithms'
by N. Krasnogor and J. Smith, IEEE Trans. on Evolutionary Computation, 2005.

Contents

Revision

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?
- What is the problem with local search methods? What are the possibilities to solve it? Name other single-state methods and describe their principles.

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?
- What is the problem with local search methods? What are the possibilities to solve it? Name other single-state methods and describe their principles.
- What is the difference between *single-state* and *population-based* methods? What is the benefit of using a population?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?
- What is the problem with local search methods? What are the possibilities to solve it? Name other single-state methods and describe their principles.
- What is the difference between *single-state* and *population-based* methods? What is the benefit of using a population?
- Describe a simple EA. What are its main components?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?
- What is the problem with local search methods? What are the possibilities to solve it? Name other single-state methods and describe their principles.
- What is the difference between *single-state* and *population-based* methods? What is the benefit of using a population?
- Describe a simple EA. What are its main components?
- When should I use an EA? When should I use a hill-climber?

Optimization, black-box, local-search, EAs

- What is *optimization*? Give some examples of optimization tasks.
- In what courses did you meet optimization?
- What *sorts of optimization tasks* do you know? What are their characteristics?
- What is the difference between *exact methods* and *heuristics*?
- What is the difference between *constructive* and *improving (generative)* methods?
- What is the *black-box optimization*? What can you do to solve such problems?
- What is the difference between *local* and *global* search?
- What is *hill-climbing*? How does it work? When does it work? What is the difference between *first-improving* and *best-improving* strategy?
- What is the problem with local search methods? What are the possibilities to solve it? Name other single-state methods and describe their principles.
- What is the difference between *single-state* and *population-based* methods? What is the benefit of using a population?
- Describe a simple EA. What are its main components?
- When should I use an EA? When should I use a hill-climber?
- Do you know other *metaheuristics*?

Hybrid Optimization Algorithms

Optimization Algorithms

Single-state (single-solution) methods:

- local search (LS)
- greedy heuristic (GH)
- simulated annealing (SA)
- tabu search (TS)
- . . .

Population-based methods:

- genetic and evolutionary algorithms (GA, EA, ES, EP, GP, . . .)
- ant colonies (ACO)
- particle swarm optimization (PSO)
- scatter search (SS)
- . . .

Hybridization and Classification of the approaches

Hybrid algorithm:

- a combination of (at least some features of) at least 2 of the above methods

Hybridization and Classification of the approaches

Hybrid algorithm:

- a combination of (at least some features of) at least 2 of the above methods

Why?

- The best results for many optimization problems were obtained by hybrid algorithms.

Hybridization and Classification of the approaches

Hybrid algorithm:

- a combination of (at least some features of) at least 2 of the above methods

Why?

- The best results for many optimization problems were obtained by hybrid algorithms.

Talbi [Tal02] classifies hybrid approaches based on several independent design issues:

- Low-level vs. high-level hybrid:
  - In **low-level hybrid**, a particular function of one algorithm is replaced by another optimization algorithm.
  - In **high-level hybrid**, the different optimization algorithms are combined without changing their inner workings.
- Relay vs. teamwork hybrid:
  - In **relay hybridization**, the individual algorithms are applied one after another, each using the output of the previous algorithm as its input.
  - In **teamwork hybridization**, each algorithm performs the search independently of the others.

[Tal02] E. G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(5):541–564, 2002.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

HRH (High-level Relay Hybrid): Several optimization algorithms executed in a sequence.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

HRH (High-level Relay Hybrid): Several optimization algorithms executed in a sequence.

- LS builds init. population for EA. The results of EA are processed by TS.
- Iterated local search, ILS.
- Variable neighborhood search, VNS.
- Greedy randomized adaptive search procedure, GRASP.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

HRH (High-level Relay Hybrid): Several optimization algorithms executed in a sequence.

- LS builds init. population for EA. The results of EA are processed by TS.
- Iterated local search, ILS.
- Variable neighborhood search, VNS.
- Greedy randomized adaptive search procedure, GRASP.

HTH (High-level Teamwork Hybrid): Several optimization algorithms executed in parallel, cooperating to find the optimum.

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

HRH (High-level Relay Hybrid): Several optimization algorithms executed in a sequence.

- LS builds init. population for EA. The results of EA are processed by TS.
- Iterated local search, ILS.
- Variable neighborhood search, VNS.
- Greedy randomized adaptive search procedure, GRASP.

HTH (High-level Teamwork Hybrid): Several optimization algorithms executed in parallel, cooperating to find the optimum.

- Island model of parallel GAs (one of the next lectures).

The 4 Main Groups of Hybrids

LRH (Low-level Relay Hybrid): An optimization method is embedded into a single-solution algorithm.

- LS embedded in SA. (What is the difference compared to a pure SA?)

LTH (Low-level Teamwork Hybrid): An optimization method is embedded into a population-based alg.

- LS embedded in EA. (Memetic algorithm, see below.)
- Greedy constructive heuristic in ACO.

HRH (High-level Relay Hybrid): Several optimization algorithms executed in a sequence.

- LS builds init. population for EA. The results of EA are processed by TS.
- Iterated local search, ILS.
- Variable neighborhood search, VNS.
- Greedy randomized adaptive search procedure, GRASP.

HTH (High-level Teamwork Hybrid): Several optimization algorithms executed in parallel, cooperating to find the optimum.

- Island model of parallel GAs (one of the next lectures).

The classification is not always unambigous, it depends on personal view.

- Is GRASP a local search with its initialization phase replaced by a constructive heuristic (LRH), or
- is it a sequence of 2 self-contained algorithms (HRH)?

Other Attributes of Hybrid Algorithms

Again, following Talbi [Tal02]:

- Homogeneous vs. heterogeneous:
  - In a **homogeneous hybrid**, all the combined algorithms are of the same type. (They may have different parameters.)
  - In a **heterogeneous hybrid**, each algorithm may be of different type.

Other Attributes of Hybrid Algorithms

Again, following Talbi [Tal02]:

- Homogeneous vs. heterogeneous:
  - In a **homogeneous hybrid**, all the combined algorithms are of the same type. (They may have different parameters.)
  - In a **heterogeneous hybrid**, each algorithm may be of different type.

- Global vs. partial:
  - In a **global hybrid**, all algorithms search the whole space.
  - In a **partial hybrid**, the problem is decomposed into subproblems and each algorithm is dedicated to the search in its own subspace. (See also the lecture on coevolution.)

Other Attributes of Hybrid Algorithms

Again, following Talbi [Tal02]:

- Homogeneous vs. heterogeneous:
  - In a **homogeneous hybrid**, all the combined algorithms are of the same type. (They may have different parameters.)
  - In a **heterogeneous hybrid**, each algorithm may be of different type.
- Global vs. partial:
  - In a **global hybrid**, all algorithms search the whole space.
  - In a **partial hybrid**, the problem is decomposed into subproblems and each algorithm is dedicated to the search in its own subspace. (See also the lecture on coevolution.)
- General vs. specialist:
  - In a **general hybrid**, all the algorithms solve the same target optimization problem.
  - In a **specialist hybrid**, some of the algorithms may solve a different optimization problem, helping the others to solve the original one.

[Tal02] E. G. Talbi. A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, 8(5):541–564, 2002.

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.
- In each node, initial population is generated using a greedy heuristic (GH)
- ...and then a genetic algorithm (GA) was run
- ...with an embedded local search (LS) improving the results each generation.

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.
- In each node, initial population is generated using a greedy heuristic (GH)
- . . . and then a genetic algorithm (GA) was run
- . . . with an embedded local search (LS) improving the results each generation.

Example: HRH(HRH(GH + LS) + LTH(GA(HRH(GH + LS))))

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.
- In each node, initial population is generated using a greedy heuristic (GH)
- ... and then a genetic algorithm (GA) was run
- ... with an embedded local search (LS) improving the results each generation.

Example: HRH(HRH(GH + LS) + LTH(GA(HRH(GH + LS))))

- HRH(GH + LS) is actually the GRASP algorithm, is it clearer now?

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.
- In each node, initial population is generated using a greedy heuristic (GH)
- . . . and then a genetic algorithm (GA) was run
- . . . with an embedded local search (LS) improving the results each generation.

Example: HRH(HRH(GH + LS) + LTH(GA(HRH(GH + LS))))

- HRH(GH + LS) is actually the GRASP algorithm, is it clearer now?
- No? What about HRH(GRASP + LTH(GA(GRASP)))?

Hybridization Design Grammar

```
<hybrid-opt-algorithm> ::= <hierarchical><flat>
<hierarchical> ::= <LRH> | <LTH> | <HRH> | <HTH>
<LRH> ::= LRH(<metaheuristic>(<metaheuristic>))
<LTH> ::= LTH(<metaheuristic>(<metaheuristic>))
<HRH> ::= HRH(<metaheuristic> + <metaheuristic>)
<HTH> ::= HTH(<metaheuristic>)
        | HTH(<metaheuristic>, <metaheuristic>)
<flat> ::= (<nature>, <optimization>, <function>)
<nature> ::= homogeneous | heterogeneous
<optimization> ::= global | partial
<function> ::= general | specialist
<metaheuristic> ::= <hybrid-opt-algorithm>
                  | LS | TS | SA | GA | ES | GP | GH | ACO | SS | ...
```

Example: HTH(HRH(GH + LTH(GA(LS))))

- Island model containing HRH(GH + LTH(GA(LS))) in each node.
- In each node, initial population is generated using a greedy heuristic (GH)
- …and then a genetic algorithm (GA) was run
- …with an embedded local search (LS) improving the results each generation.

Example: HRH(HRH(GH + LS) + LTH(GA(HRH(GH + LS))))

- HRH(GH + LS) is actually the GRASP algorithm, is it clearer now?
- No? What about HRH(GRASP + LTH(GA(GRASP)))?
- GRASP creates the initial population for a memetic algorithm that uses GRASP to improve solutions of GA.

Memetic Algorithms

Memetic Algorithms

Memetic algorithm (MAs):

- hybrid of evolutionary algorithm with embedded local search, i.e.
- EA that includes one or more LS phases within its evolutionary cycle.
- MA is **LTH(EA(LS))** using Talbi's taxonomy.

Memetic Algorithms

Memetic algorithm (MAs):

- hybrid of evolutionary algorithm with embedded local search, i.e.
- EA that includes one or more LS phases within its evolutionary cycle.
- MA is **LTH(EA(LS))** using Talbi's taxonomy.

EA vs. Local-search:

- *Local search* quickly identifies an optimum, but the optimum is often only local
- *EA* is more robust against getting stuck in local optima; it uses large population and searches several places of the search space in parallel. It is thus *slower*.

Memetic Algorithms

Memetic algorithm (MAs):

- hybrid of evolutionary algorithm with embedded local search, i.e.
- EA that includes one or more LS phases within its evolutionary cycle.
- MA is **LTH(EA(LS))** using Talbi's taxonomy.

EA vs. Local-search:

- *Local search* quickly identifies an optimum, but the optimum is often only local
- *EA* is more robust against getting stuck in local optima; it uses large population and searches several places of the search space in parallel. It is thus *slower*.

When implemented well,

- MAs are faster than ordinary EAs, and
- MAs are more robust against getting stuck in local optima than local search.

Memetic Algorithms

Memetic algorithm (MAs):

- hybrid of evolutionary algorithm with embedded local search, i.e.
- EA that includes one or more LS phases within its evolutionary cycle.
- MA is **LTH(EA(LS))** using Talbi's taxonomy.

EA vs. Local-search:

- *Local search* quickly identifies an optimum, but the optimum is often only local
- *EA* is more robust against getting stuck in local optima; it uses large population and searches several places of the search space in parallel. It is thus *slower*.

When implemented well,

- MAs are faster than ordinary EAs, and
- MAs are more robust against getting stuck in local optima than local search.

Talbi's taxonomy ignores many MA design issues.

Models of Evolution

Darwinian evolution

- the whole "plan" of an organism is inscribed in its genes
- the survival of the organism depends on its genetic constitution only
- used in ordinary EAs

Models of Evolution

Darwinian evolution

- the whole "plan" of an organism is inscribed in its genes
- the survival of the organism depends on its genetic constitution only
- used in ordinary EAs

Lamarckian evolution

- characteristics acquired during the organism's lifetime can be transferred to the offspring
- used in Lamarckian MAs:
  - an individual in EA is improved by a local search
  - the improved individual replaces the original individual

Models of Evolution

Darwinian evolution

- the whole “plan” of an organism is inscribed in its genes
- the survival of the organism depends on its genetic constitution only
- used in ordinary EAs

Lamarckian evolution

- characteristics acquired during the organism’s lifetime can be transferred to the offspring
- used in Lamarckian MAs:
  - an individual in EA is improved by a local search
  - the improved individual replaces the original individual

Baldwin effect

- the genetic information encodes the ability to learn
- organisms able to learn have higher chance of survival, which in turn allows the “right” genes to survive and breed
- used in Baldwinian MAs:
  - an individual in EA is improved by a local search
  - the original individual is preserved in the population, but its fitness is changed to the fitness of the improved individual

MA: Main Design Issues

What is the best trade-off between local-search and the global search provided by evolution?

MA: Main Design Issues

What is the best trade-off between local-search and the global search provided by evolution?

- Where (in which phase of the evolutionary cycle) and when (in which evolutionary cycle) should the local search be applied?
- Which individuals should be improved? (The best ones? All? Random ones?)
- How long should each local search be allowed to run? (Make a complete run? Only perform one step of LS?)
- How can the genetic operators be best combined (modified) with local search to achieve the synergy?
- Should we use the Lamarckian or Baldwinian model?

MAs Classification

Krasnogor and Smith [KS05] introduce the concept of *schedulers*:

- *Fine-grained mutation scheduler* fS_M replaces the mutation operator and organizes the mutation and LS.
- *Fine-grained crossover scheduler* fS_R replaces the crossover operator and organizes the crossover and LS.
- *Coarse-grained scheduler* cS replaces selection and replacement, and organizes the interplay between selection, crossover and LS. It can supply the population statistics to LS.
- *Meta scheduler* mS provides a historical information to LS inside MA.

MAs Classification

Krasnogor and Smith [KS05] introduce the concept of *schedulers*:

- *Fine-grained mutation scheduler* fS_M replaces the mutation operator and organizes the mutation and LS.
- *Fine-grained crossover scheduler* fS_R replaces the crossover operator and organizes the crossover and LS.
- *Coarse-grained scheduler* cS replaces selection and replacement, and organizes the interplay between selection, crossover and LS. It can supply the population statistics to LS.
- *Meta scheduler* mS provides a historical information to LS inside MA.

A 4-bit number can then be assigned to each MA:

$$D = (b_4, b_3, b_2, b_1)$$

- $b_1 = 1 \iff$ MA contains fS_M.
- $b_2 = 1 \iff$ MA contains fS_R.
- $b_3 = 1 \iff$ MA contains cS.
- $b_4 = 1 \iff$ MA contains mS.

MAs Classification

Krasnogor and Smith [KS05] introduce the concept of *schedulers*:

- *Fine-grained mutation scheduler* fS_M replaces the mutation operator and organizes the mutation and LS.
- *Fine-grained crossover scheduler* fS_R replaces the crossover operator and organizes the crossover and LS.
- *Coarse-grained scheduler* cS replaces selection and replacement, and organizes the interplay between selection, crossover and LS. It can supply the population statistics to LS.
- *Meta scheduler* mS provides a historical information to LS inside MA.

A 4-bit number can then be assigned to each MA:

$$D = (b_4, b_3, b_2, b_1)$$

- $b_1 = 1 \Longleftrightarrow$ MA contains fS_M.
- $b_2 = 1 \Longleftrightarrow$ MA contains fS_R.
- $b_3 = 1 \Longleftrightarrow$ MA contains cS.
- $b_4 = 1 \Longleftrightarrow$ MA contains mS.

Higher value of D implies more complex algorithm, but not necessarily a better one!

[KS05] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *Evolutionary Computation, IEEE Transactions on*, 9(5):474–488, 2005.

Which LS should I choose?

It depends.

- Even within a single problem class, e.g. TSP, different LS operators are best for different instances of TSP. The choice is *instance-specific*.
- Even during one optimization run on one problem instance, different LS operators are best for different phases of evolution. The choice is *time-dependent*.

Which LS should I choose?

It depends.

- Even within a single problem class, e.g. TSP, different LS operators are best for different instances of TSP. The choice is *instance-specific*.
- Even during one optimization run on one problem instance, different LS operators are best for different phases of evolution. The choice is *time-dependent*.

Based on these observations, a sensible approach could be:

- do not decide a priori which LS should be used;
- incorporate several of them and consider methods to avoid spending time using the unproductive ones.
- This implies at least some way of adapting of operator probabilities, which in turn implies the presence of cS or mS.

How should I best integrate LS into EA?

It depends. We can list some suggestions:

How should I best integrate LS into EA?

It depends. We can list some suggestions:

Lamarckian vs. Baldwinian:

- Lamarckian learning can happen before or after the application of other operators.
- There is a little point in applying Baldwinian search after parent selection, but before recombination or mutation.

How should I best integrate LS into EA?

It depends. We can list some suggestions:

Lamarckian vs. Baldwinian:

- Lamarckian learning can happen before or after the application of other operators.
- There is a little point in applying Baldwinian search after parent selection, but before recombination or mutation.

If Lamarckian search is continued to optimality:

- recombination and mutation will likely produce offspring worse than parents.
- We hope that the offspring will be in the basin of attraction of high-quality LO.
- To prevent selection discarding these new worse offspring, perform LS on them prior to selection.

How should I best integrate LS into EA?

It depends. We can list some suggestions:

Lamarckian vs. Baldwinian:

- Lamarckian learning can happen before or after the application of other operators.
- There is a little point in applying Baldwinian search after parent selection, but before recombination or mutation.

If Lamarckian search is continued to optimality:

- recombination and mutation will likely produce offspring worse than parents.
- We hope that the offspring will be in the basin of attraction of high-quality LO.
- To prevent selection discarding these new worse offspring, perform LS on them prior to selection.

Lamarckian approach is used most often, the LS is continued to optimality.

- This leads to the loss of diversity very quickly, MA gets stuck in LO, or stagnates on a plateau.
- A cS can monitor the population statistics and re-introduce diversity.

How should I set the global-local tradeoff?

The majority of MAs apply LS to every individual in every generation of the EA. But a cS can be used to

- choose individuals to be optimized by LS,
- set the intensity of LS,
- set the probability of applying LS.

How should I set the global-local tradeoff?

The majority of MAs apply LS to every individual in every generation of the EA. But a cS can be used to

- choose individuals to be optimized by LS,
- set the intensity of LS,
- set the probability of applying LS.

Example of an approach to select points for LS:

- Single-linkage clustering is used to prevent running LS too often in the same basin of attraction.
- Each basin is represented by a set of points used for LS.
- If a point is used in LS, it is assigned to the respective cluster.
- The number of clusters grows, the number of points in the cluster grows as well (if not limited somehow).
- A candidate for LS is not used, if it lies within a critical distance from an already clustered point.

Summary

- Hybrid optimization algorithm is virtually any combination of 2 or more other algorithms.
  - Many existing algorithms can be described as hybrids.
  - Using the systematic taxonomy, many of non-existing hybrids can be easily created. (This does not mean they will be better than the existing ones.)
- MAs are a broad class of hybrid optimization algorithms combining
  - the robustness of evolutionary algorithms and
  - the speed of local search methods.
- Their design is
  - simple from a practitioner's point of view (very often they just work), but
  - becomes harder when we want to exploit the most of it (extensive testing and comparisons needed), and is
  - incredibly complex from the theoretical point of view when the exact interplay between particular algorithm features should be undestood or modeled.