Model-Based Evolutionary Algorithms
Part 2: Linkage Tree Genetic Algorithm

Dirk Thierens

Universiteit Utrecht
The Netherlands

Joint work: Peter Bosman, CWI Amsterdam

MBEA

Evolutionary Algorithms

- Population-based, stochastic search algorithms
- Exploitation: selection
- Exploration: mutation & crossover

Model-Based Evolutionary Algorithms

- Population-based, stochastic search algorithms
- Exploitation: selection
- Exploration:
  1. Learn a (probabilistic) model from selected solutions
  2. Generate new solutions from the model (& population)

GOMEA

Gene-pool Optimal Mixing Evolutionary Algorithm

- Population-based, stochastic search algorithms
- Exploitation: selection (by replacement)
- Exploration:
  1. Learn a Family-Of-Subsets model
  2. Generate new solutions through optimal mixing

GOMEA: design objectives

1. Be able to efficiently learn dependency information (= linkage) between variables
2. Be able to efficiently decide between competing building blocks
3. Transfer all optimal building blocks from the parents to the offspring solution

Family Of Subsets (FOS) model

- Key idea is to identify groups of problem variables that together make an important contribution to the quality of solutions.
- These variable groups are interacting in a non-linear way and should be processed as a block = building block

FOS $\mathcal{F}$

Dependency structure generally called a Family Of Subsets (FOS).

- Let there be ℓ problem variables $x_0, x_1, \ldots, x_{\ell-1}$.
- Let S be a set of all variable indices $\{0, 1, \ldots, \ell - 1\}$.
- A FOS $\mathcal{F}$ is a set of subsets of the set S.
- FOS $\mathcal{F}$ is a subset of the powerset of S ($\mathcal{F} \subseteq \mathcal{P}(S)$).

Example Family Of Subsets (FOS) models:

- Univariate FOS structure

$$\mathcal{F} = \{\{0\},\{1\},\{2\},\{3\},\{4\},\{5\},\{6\},\{7\},\{8\},\{9\}\}$$

- Marginal Product FOS Structure

$$\mathcal{F} = \{\{0,1,2\},\{3\},\{4,5\},\{6,7,8,9\}\}$$

- Linkage Tree FOS Structure

$$\mathcal{F} = \{\{7,5,8,6,9,0,3,2,4,1\},$$

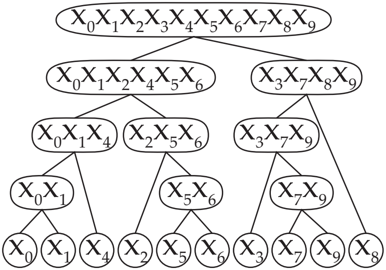
$$\{7,5,8,6,9\},\{0,3,2,4,1\},\{7\},\{5,8,6,9\},$$

$$\{0,3,2,4\},\{1\},\{5,8,6\},\{9\},\{0,3\},\{2,4\},$$

$$\{5,8\},\{6\},\{0\},\{3\},\{2\},\{4\},\{5\},\{8\}\}$$

Linkage Tree

- Problem variables in subset are considered to be dependent on each other but become independent in a child subset.
- ≈ Path through dependency space, from univariate to joint.
- Linkage tree has ℓ leaf nodes (= single problem variables) and $\ell - 1$ internal nodes.



Linkage Tree Learning

- Start from univariate structure.
- Build linkage tree using bottom-up hierarchical clustering algorithm.
- Similarity measure:
  1. Between individual variables X and Y: mutual information $I(X,Y)$.

$$I(X,Y) = H(X) + H(Y) - H(X,Y)$$

  2. Between cluster groups X_{F^i} and X_{F^j}: average pairwise linkage clustering (= unweighted pair group method with a arithmetic mean: UPGMA).

$$I^{UPGMA}(X_{F^i}, X_{F^j}) = \frac{1}{|X_{F^i}||X_{F^j}|} \sum_{X \in X_{F^i}} \sum_{Y \in X_{F^j}} I(X,Y).$$

($H(X), H(Y), H(X,Y)$ are the marginal and joint entropies)

Linkage Tree Learning

- This agglomerative hierarchical clustering algorithm is computationally efficient.
- Only the mutual information between pairs of variables needs to be computed once, which is a $O(\ell^2)$ operation.
- The bottom-up hierarchical clustering can also be done in $O(\ell^2)$ computation by using the *reciprocal nearest neighbor chain* algorithm.
- note: commonly used bottom-up hierarchical clustering algorithms (*hclust* and *agnes* in R) have $O(\ell^3)$ complexity.

Optimal Mixing EA

- FOS linkage models specify the linked variables.
- A subset of the FOS is used as crossover mask
- Crossover is greedy: only improvements (or equal) are accepted.
- Each generation a new FOS model is build from selected solutions.
- For each solution in the population, all subsets of the FOS are tried with a donor solution randomly picked from the population
- Recombinative OM (ROM) and Gene-pool OM (GOM)
  - ROM is GA-like: select single donor solution to perform OM with
  - GOM is EDA-like: select new donor solution for each subset in OM

Gene-pool Optimal Mixing EA

```
GOMEA()
   Pop ← InitPopulation()
   while NotTerminated(Pop)
      FOS ← BuildFOS(Pop)
      forall Sol ∈ Pop
         forall SubSet ∈ FOS
            Donor ← Random(Pop)
            Sol ← GreedyRecomb(Sol,Donor,Subset,Pop)
   return Sol

GreedyRecomb(Sol,Donor,SubSet,Pop)
   NewSol ← ReplaceSubSetValues(Sol,SubSet,Donor)
      if ImprovementOrEqual(NewSol,Sol)
         then Sol ← NewSol
   return Sol
```

Recombinative Optimal Mixing EA

```
ROMEA()
   Pop ← InitPopulation()
   while NotTerminated(Pop)
      FOS ← BuildFOS(Pop)
      forall Sol ∈ Pop
         Donor ← Random(Pop)
         forall SubSet ∈ FOS
            Sol ← GreedyRecomb(Sol,Donor,Subset,Pop)
   return Sol

GreedyRecomb(Sol,Donor,SubSet,Pop)
   NewSol ← ReplaceSubSetValues(Sol,SubSet,Donor)
      if ImprovementOrEqual(NewSol,Sol)
         then Sol ← NewSol
   return Sol
```

Optimal Mixing

- Characteristic of Optimal Mixing is the use of intermediate function evaluations (inside variation)
- Can be regarded as greedy improvement of existing solutions
- Coined Optimal Mixing because better instances for substructures are immediately accepted and not dependent on noise coming from other parts of the solution
- Building block competition no longer a stochastic decision making problem that requires a sizable minimal population size
- Population sizes in GOMEA much smaller than in GAs or EDAs.

Linkage Tree Genetic Algorithm

- The LTGA is an instance of GOMEA that uses a Linkage Tree as FOS model
- Each generation a new hierarchical cluster tree is build.
- For each solution in population, traverse tree starting at the top.
- Nodes (= clusters) in the linkage tree used as crossover masks.
- Select random donor solution, and its values at the crossover mask replace the variable values from the current solution.
- Evaluate new solution and accept if better/equal, otherwise reject.

Convergence model

Univariate FOS model on onemax problem

- ℓ: string length
- n: population size
- $p(t)$: proportion bit '1' at generation t
- $q(t)$: proportion bit '0' at generation t

Bit '0' only survive if parent and donor both have a '0' at that index:

- $q(t+1) = q^2(t)$
- $p(t) = 1 - [1 - p(0)]^{2^t}$

Number of function evaluations FE:

- In 1 generation:

$$FE = 2\, p(t)([1 - p(t)] \times \ell \times n$$

- After g generations:

$$FE = \sum_{t=0}^{g} 2\, p(t)([1 - p(t)] \times \ell \times n$$

- After convergence g_{conv}:

$$FE = 2\, [1 - p(0)] \times \ell \times n$$

- Initial random population ($p(0) = 0.5$):

$$\boxed{FE = \ell \times n} \quad \Rightarrow \quad \boxed{O(\ell \, \log \ell)}$$

$$
\begin{aligned}
& \sum_{t=0}^{g} p(t)([1-p(t)] \\
= & \sum_{t=0}^{g} q(t)([1-q(t)] \\
= & \ q(0)[1-q(0)] + q(1)[1-q(1)] + \cdots + q(g)[(1-q(g)] \\
= & \ q(0) - q(1) + q(1) - q(2) + \cdots - q(g) + q(g) - q^2(g) \\
= & \ q(0) - [q(0)]^{2^{(g+1)}} \\
g_{conv}: \rightarrow & \ q(0)
\end{aligned}
$$

Minimal population size

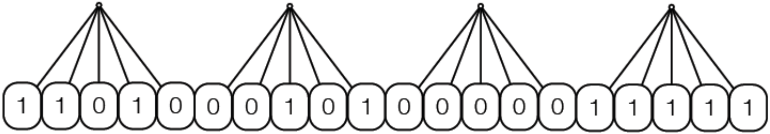
Need to have at least one bit '1' at each index:

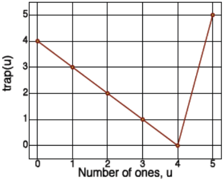
$$
\begin{aligned}
Prob[success] &= [1 - (1 - p(0))^n]^\ell \\
&\approx 1 - \ell\,[1 - p(0)]^n \\
1 - 0.01 &= 1 - \ell\,[1 - \frac{1}{2}]^n \\
n &= \log_2(100\ell) \\
n &= O(\log \ell)
\end{aligned}
$$

Deceptive Trap Function

Interacting, non-overlapping, deceptive groups of variables.

$$f_{\text{DT}}(x) = \sum_{i=0}^{l-k} f_{\text{DT}}^{\text{sub}}\left(x_{(i,\ldots,i+k-1)}\right)$$



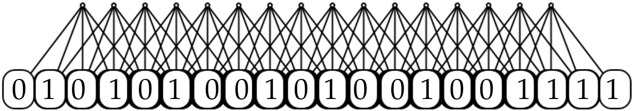


Nearest-neighbor NK-landscape

- Overlapping, neighboring random subfunctions

$$f_{\text{NK-S1}}(x) = \sum_{i=0}^{l-k} f_{\text{NK}}^{\text{sub}}\left(x_{(i,\ldots,i+k-1)}\right) \quad \text{with } f_{\text{NK}}^{\text{sub}}\left(x_{(i,\ldots,i+k-1)}\right) \in [0..1]$$

- eg. 16 subsfcts, length $k = 5$, overlap $o = 4 \Rightarrow$ stringlength $\ell = 20$

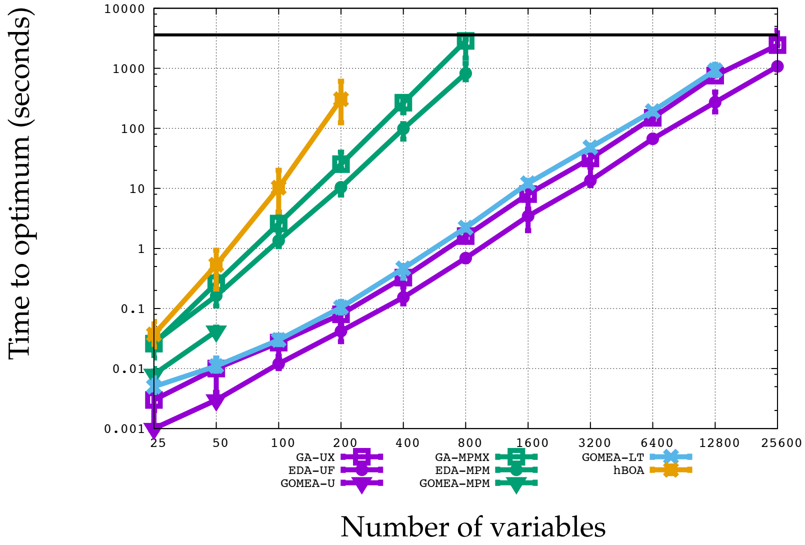


- Global optimum computed by dynamic programming
- Benchmark function: structural information is not known !
- $\Rightarrow$ Randomly shuffled variable indices.

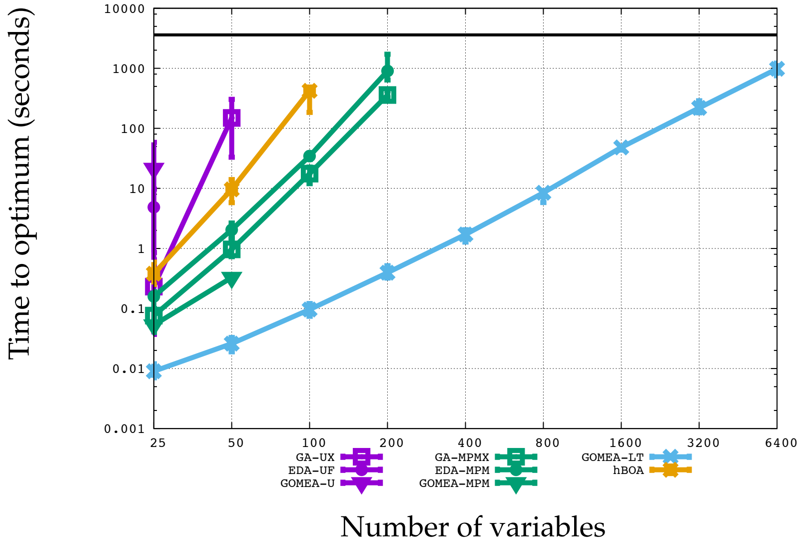
Experiments

- Compare GA, EDA, and GOMEA while each are learning the Marginal Product (MP) *FOS* structure, and GOMEA learning the Linkage Tree (LT) as *FOS* structure.
- Note:
  - EDA using MP = Extended Compact GA (ECGA).
  - GOMEA using LT = Linkage Tree Genetic Algorithm (LTGA).
  - hBOA = EDA learning a Bayesian network each generation.

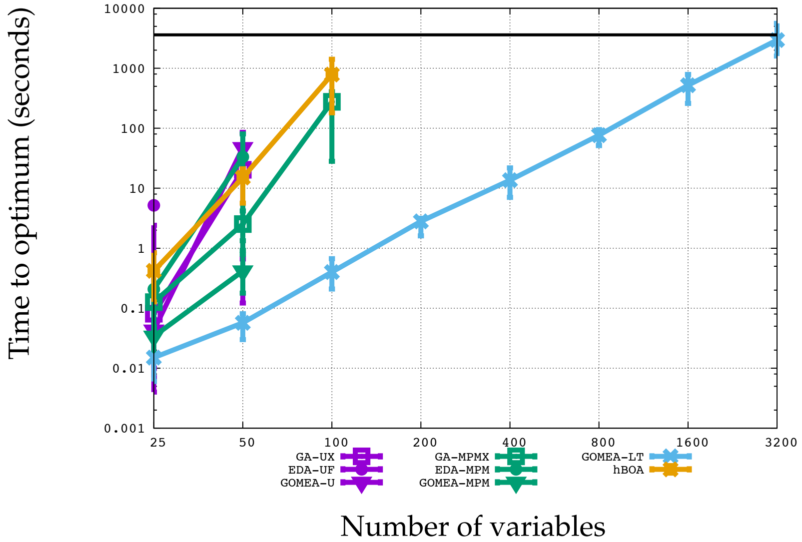
Experiments - Onemax



Experiments - Deceptive trap



Experiments - Overlapping NK



Experiments

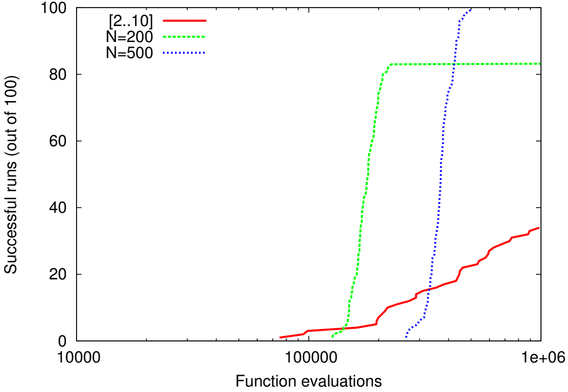
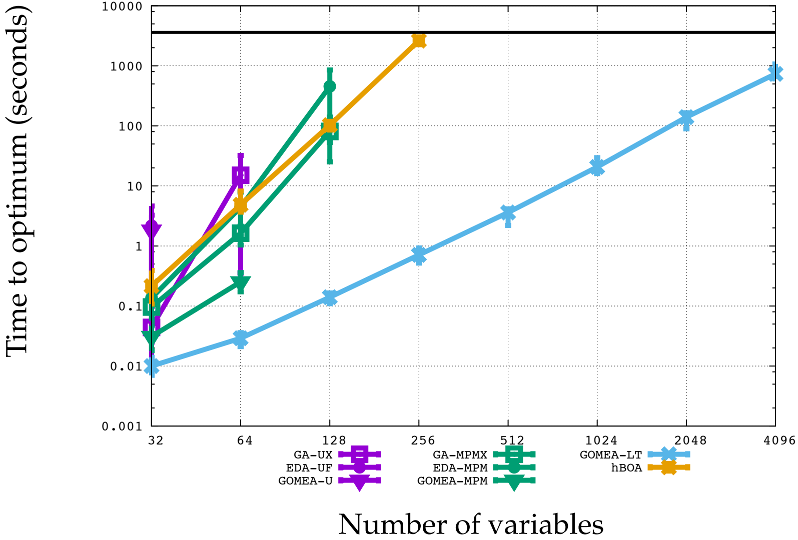


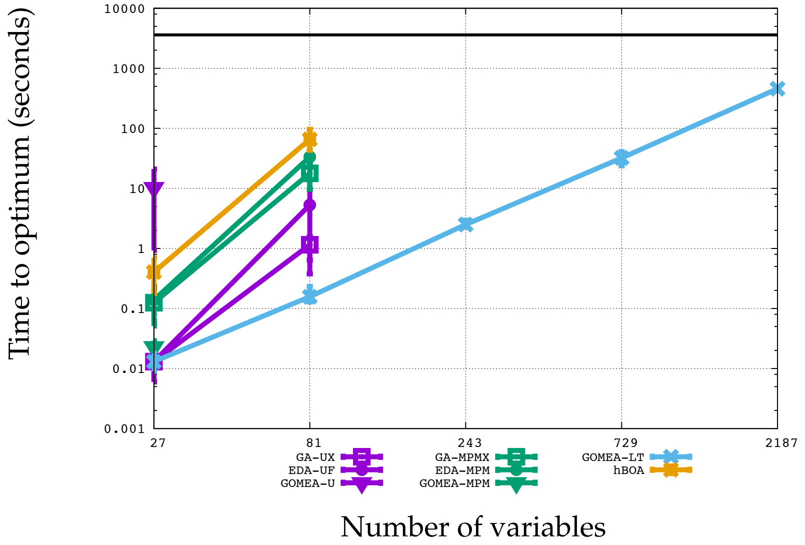
Figure: LTGA vs. ILS: 100 NK problems

Iterated Local Search: perturbation size each time randomly picked between 2 and 10 bits (= better than any fixed value).

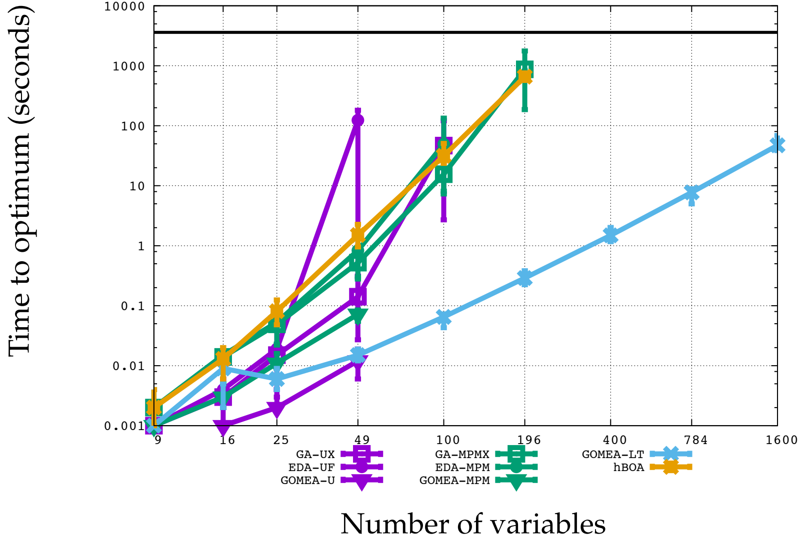
Experiments - HIFF



Experiments - HTrap



Experiments - MAX-CUT 2D square grid



Conclusions[1]

- "Blind" Evolutionary Algorithms are limited in their capability to detect and exploit partial solutions (building blocks).
- Optimal Mixing Evolutionary Algorithms efficiently learn important building blocks and efficiently decide between competing building blocks
- Linkage Tree appears to be good compromise between FOS model complexity and search efficiency.

[1] http://homepages.cwi.nl/~bosman/source_code.php