

# Loopy Substructural Local Search for the Bayesian Optimization Algorithm

Claudio F. Lima<sup>1</sup>   Martin Pelikan<sup>2</sup>   Fernando G. Lobo<sup>1</sup>  
David E. Goldberg<sup>3</sup>

<sup>1</sup>University of Algarve, Portugal

<sup>2</sup>University of Missouri at St. Louis, USA

<sup>3</sup>University of Illinois at Urbana-Champaign, USA

SLS 2009, Brussels, Belgium

# Motivation

- Estimation of distribution algorithms (EDAs) can solve many challenging problems in an efficient and scalable manner.
- Although effective at exploring the search space, they inherit a common drawback from traditional global search methods.
  - Slower convergence to optimal solutions when compared to appropriate local searchers.
- This observation often leads to the combination of global and local search approaches.
- EDA research lacks on methods for designing and combining competent global and local-search methods.
  - That can exploit problem decomposition.

# Outline

## Topics

- 1 Bayesian Optimization Algorithm (BOA).
- 2 Substructural Local Search in BOA.
- 3 Loopy Belief Propagation in Graphical Models.
- 4 Loopy Substructural Local Search in BOA.
- 5 Results and Discussion.
- 6 Summary and Conclusions.

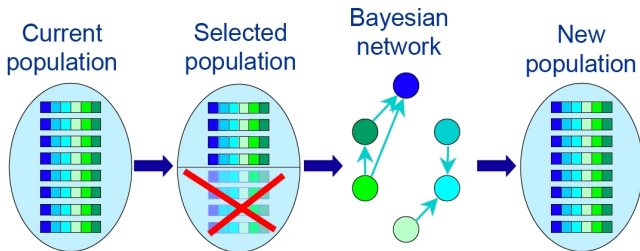
# Estimation of Distribution Algorithms

## EDAs

- Population-based search and optimization procedures.
- Population of promising solutions is modeled to generate new candidate solutions.
- “Similar” to genetic algorithms: replace genetic operators by building and sampling from a probabilistic model (PM).
- PMs used can range from simple univariate probability vectors to more complex Markov or Bayesian networks.
- Popular EDAs:
  - Population-based incremental learning (PBIL).
  - Extended compact genetic algorithm (ECGA).
  - Bayesian optimization algorithm (BOA).

# Bayesian Optimization Algorithm (BOA)

- Use Bayesian networks (BNs) to represent conditional dependencies between variables.



$$p(\mathbf{X}) = p(X_1 | X_5 X_6) p(X_6 | X_5) p(X_5) p(X_3 | X_2 X_4) p(X_2) p(X_4)$$

# Model Learning in BOA

BOA encodes the following joint probability distribution

$$p(X) = \prod_{i=1}^{\ell} p(X_i | \Pi_i), \quad (1)$$

where  $X = (X_1, X_2, \dots, X_\ell)$  are problem variables,  $\Pi_i$  is the set of parent nodes of  $X_i$ , and  $p(X_i | \Pi_i)$  is the conditional probability of  $X_i$  given its parents  $\Pi_i$ .

Appropriate BN structure is learned at each generation

- A simple learning algorithm is used.
  - Start with a simple model structure (no dependencies).
  - Increase model complexity while the likelihood of the model w.r.t. the data increases.

# Modeling Fitness in BOA

## Bayesian networks extended to model solution quality

- The probabilistic model can also estimate the fitness of a solution:

$$f_{est}(X_1, X_2, \dots, X_\ell) = \bar{f} + \sum_{i=1}^{\ell} (\bar{f}(X_i|\Pi_i) - \bar{f}(\Pi_i)), \quad (2)$$

where  $\bar{f}$  is the average fitness of all solutions used to learn the surrogate,  $\bar{f}(X_i|\Pi_i)$  is the average fitness of solutions with  $X_i$  and  $\Pi_i$ , and  $\bar{f}(\Pi_i)$  is the average fitness of all solutions with  $\Pi_i$ .

- Surrogate fitness model is composed by:
  - Structure (BN structure).
  - Parameters (schema average fitness, etc).

# Model-Based Efficiency Enhancement Techniques

## EDA's great advantage over other methods

- Probabilistic modeling in EDAs allow to identify patterns in good solutions.
- Learned probabilistic models provide valuable information.
  - Get a better insight about the optimization problem.
  - Speedup the search process in EDAs even more.

## Induction of neighborhoods

- The probabilistic models reveal important dependencies between variables that can be exploited in local search.

# Substructural Local Search (SLS)

## Traditional Local Search

- Typically neighborhood operators have fixed structure.
- Implicit tradeoff between generalization and efficiency.

## SLS in EDAs

- Use substructural neighborhoods to perform local search in EDAs.
- Topology of neighborhoods is defined by the learned probabilistic models.
- Exploit underlying problem structure while not losing (so much) generality of application.

# Substructural Local Search in BOA

## A simple SLS method for BOA

- Each variable and its corresponding parent variables form a substructural neighborhood—*parental neighborhood*.
- For each substructural neighborhood, choose the best configuration according to the estimated fitness.
  - Within neighborhood  $(X_i, \Pi_i)$  choose the configuration with higher  $\bar{f}(X_i, \Pi_i)$ .
  - Neighborhood size of  $2^{\Pi_i+1}$  for binary representation.
- Performed at each generation after model learning.
- Applied to a proportion of the population (typically small).
- Notice however that this LS method does not take into account the context of possible overlapping interactions.

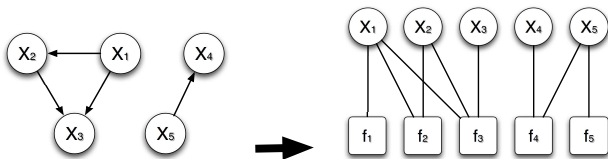
# Belief Propagation in Graphical Models

- Method for performing exact and approximate inference in graphical models.
  - ① Obtain marginal probability distributions.
  - ② Find the most probable explanation (configuration) for the graphical model.
- BP typically applied to factor graphs.
  - Unifying representation for Bayesian and Markov networks.
- Factor graphs explicitly express the factorization structure of probability distribution:

$$g(x_1, x_2, \dots, x_\ell) = \frac{1}{Z} \prod_{l \in \mathcal{F}} f_l(x_{N_l}), \quad (3)$$

where  $Z = \sum_x \prod_{l \in \mathcal{F}} f_l(x_{N_l})$ ,  $l$ : factor index,  $N_l$ : subset of variable indices associated with factor  $l$ , and factor  $f_l$ : nonnegative function.

# Belief Propagation in Factor Graphs



$$g(x_1, x_2, x_3, x_4, x_5) = \frac{1}{Z} f_1(x_1) f_2(x_1, x_2) f_3(x_1, x_2, x_3) f_4(x_4, x_5) f_5(x_5)$$

- Exchange messages between factor nodes (squares) and variable nodes (circles) until reaching a stable situation.
- BP uses message-passing algorithms.
  - Sum-product algorithm (marginal probabilities).
  - Max-product algorithm (most probable configuration).

# Messages in Belief Propagation

- Messages  $m_{i \rightarrow I}$ , sent from variables  $i \in \mathcal{V}$  to neighboring factors  $I \in N_i$ .

$$m'_{i \rightarrow I}(x_i) = \prod_{J \in N_i \setminus I} m_{J \rightarrow i}(x_i) \quad \forall i \in \mathcal{V}, \forall I \in N_i, \quad (4)$$

- Messages  $m_{I \rightarrow i}$ , sent from factors  $I \in \mathcal{F}$  to neighboring variables  $i \in N_I$ .

$$m'_{I \rightarrow i}(x_i) = \sum_{x_{N_I \setminus i}} f_I(x_{N_I}) \prod_{j \in N_I \setminus i} m_{j \rightarrow I}(x_j) \quad \forall I \in \mathcal{F}, \forall i \in N_I, \quad (5)$$

$$m'_{I \rightarrow i}(x_i) = \max_{x_{N_I \setminus i}} \left( f_I(x_{N_I}) \prod_{j \in N_I \setminus i} m_{j \rightarrow I}(x_j) \right) \quad \forall I \in \mathcal{F}, \forall i \in N_I, \quad (6)$$

# Loopy Belief Propagation (LBP)

## Message-passing

- Each node sends and receives messages from neighbors.
- Outgoing messages are functions of incoming messages.
- Messages exchanged according to some schedule until reaching stable state (messages no longer change).
- Essentially, *different factors try to reach a consensus that determines the marginal probabilities or the MPC.*

## Exact vs. approximate inference

- If graph acyclic, beliefs are exact.
- If graph cyclic, beliefs are approximate.
- BP in cyclic graphs known as **loopy** BP.

# Loopy Substructural Local Search in BOA

- Local search based on loopy belief propagation.
- Factor nodes contain estimated fitness information  $\bar{f}(X_{N_i})$ .
- Max-product algorithm applied to find MPC, which is the configuration that maximizes fitness!
- Removal of redundant factor nodes: whose variable set is a subset of another factor.
  - Possible because for instance  $\bar{f}(X_1, X_2, X_3)$  is more informative than  $\bar{f}(X_1, X_2)$ .
- In case of ties for certain variables, enumerate all possible configurations (up to  $\ell$ ) and instantiate them as different solutions.

# Loopy SLS Pseudocode

- (1) Map the current Bayesian network  $\mathbf{B}$  into a factor graph  $\mathbf{F}$ , where factor nodes store substructural fitness information  $\bar{f}(X_i, \Pi_i)$ .
- (2) Remove factor nodes (and corresponding edges) whose variable set is a subset of another factor in  $\mathbf{F}$ .
- (3) Perform loopy belief propagation in  $\mathbf{F}$ . Return the most probable configuration MPC and possible number of tied positions  $n_t$ .
- (4) **If**  $n_t = 0$ , instantiate an individual with the values from **MPC**;  
**Else If**  $2^{n_t} \leq \ell$ , enumerate all possible  $2^{n_t}$  configurations and instantiate them in  $2^{n_t}$  different individuals;  
**Else** enumerate  $\ell$  randomly chosen configurations out of  $2^{n_t}$  and instantiate them in  $\ell$  different individuals.
- (5) Evaluate the resulting individuals.

# Experimental Setup

## Test problem

- Algorithms tested on the  $m$ - $k$  trap problem.
  - Problem composed by  $m$  copies of a  $k$ -bit trap function.

$$f_{\text{trap}}(u) = \begin{cases} k, & \text{if } u = k \\ k - 1 - u, & \text{otherwise} \end{cases} \quad (7)$$

- For  $k \geq 3$  trap function is fully deceptive.
  - Any lower than  $k$ -order statistics will mislead the search away from the optimum.
- Identification and maintenance of building-blocks is crucial for success, otherwise exponential scalability.
- Problem instances used have overlap  $o = \{0, 1, 2, 3\}$  between trap functions of  $k = 5$ .

# Experimental Setup

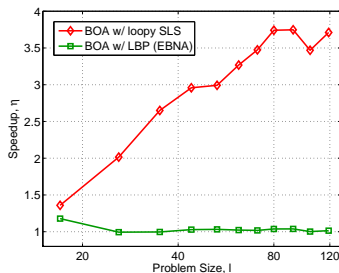
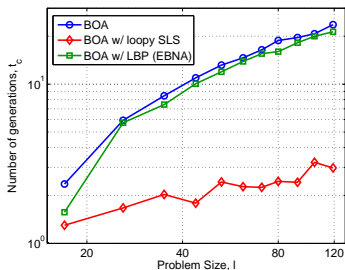
## Algorithms compared

- Standard BOA (no local search).
- BOA w/ simple SLS.
- BOA w/ loopy SLS.
- BOA w/ “standard” loopy belief propagation.
  - Factor nodes contain conditional probabilities (instead of explicit fitness information).
  - Factor graph can not be further simplified because all conditional probabilities are required to define joint probability distribution.

## Scalability results

- Analyze number of evaluations required for increasing problem instances.

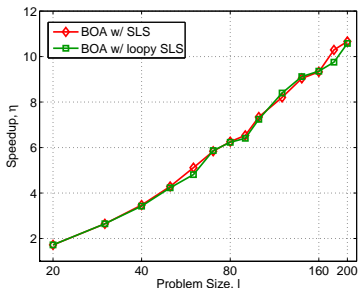
# Standard LBP vs. Loopy SLS ( $k = 5$ and $o = 2$ )



- Standard LBP (guided by conditional probabilities) marginally improves BOA results.
- Loopy SLS (guided by estimated fitness) significantly reduces num. of generations. Speedups up to 3.75.

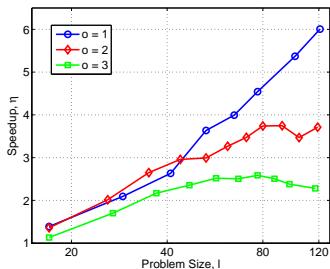
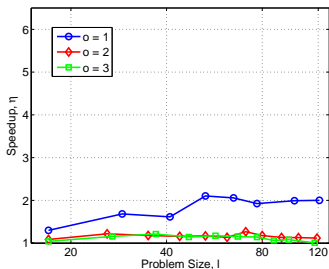
## Simple vs. Loopy SLS ( $k = 5$ and $o = 0$ )

- Problem composed by non-overlapping traps.



- Both simple and loopy SLS succeed to improve BOA.
- Speedup consistently increases for larger problem instances  $l$ .
- Scales approximately as  $\theta(\sqrt{l})$ .
- For  $l = 200$ , BOA+SLS takes an order of magnitude less function evaluations than BOA.

# Simple vs. Loopy SLS ( $k = 5$ and $o = \{1, 2, 3\}$ )



- Simple SLS fails to maintain speedups.
- Loopy SLS still provides significant speedups in evals.
- Speedup decreases with increasing overlap.
- Overlap effect is similar to that of noise.

# Summary and Conclusions

- Paper proposes a substructural local search method for the Bayesian optimization algorithm.
- Method inspired on belief propagation principles.
  - Find the best configuration of the Bayesian network based on substructural fitness information.
- Empirical results demonstrate that SLS can substantially reduce the number of function evaluations for a class of boundedly-difficult problems.
  - Providing speedups superior to 10.
  - Speedup scales with problem size as  $O(\sqrt{\ell})$ .
- Extend experiments to other classes of problems.
  - Parameters for the surrogate fitness model can be estimated with more sophisticated methods for linear estimation.