iBOA
Incremental Bayesian Optimization Algorithm

Martin Pelikan, Kumara Sastry and David E. Goldberg

Missouri Estimation of Distribution Algorithms Laboratory (MEDAL)
University of Missouri, St. Louis, MO
http://medal.cs.umsl.edu/
pelikan@cs.umsl.edu

Illinois Genetic Algorithms Laboratory
University of Illinois at Urbana-Champaign, Urbana, IL
http://www-illigal.ge.uiuc.edu/
kumara@kumarasastry.com deg@uiuc.edu
Motivation

Estimation of distribution algorithms (EDAs)

- EDAs guide search by building and sampling an explicit probabilistic model of high-quality solutions.
- Why not fully replace population by the probabilistic model?

Incremental EDAs

- Replace population by probabilistic model.
- Update the model incrementally using a few solutions.
- This can lead to savings in memory cost and speedups.
- Proof of concept: Solution of a noisy problem with more than one billion bits (Goldberg et al., 2007).
- But all incremental EDAs use univariate or bivariate models.

Purpose

- Propose incremental Bayesian optimization algorithm.
Outline

1. Starting points
   - Bayesian optimization algorithm (BOA).
   - Incremental EDAs: Compact genetic algorithm (cGA).

2. Incremental BOA (iBOA).

3. Experiments.

4. Future work.

5. Summary and conclusions.
Bayesian Optimization Algorithm (BOA)

BOA (Pelikan, Goldberg, and Cantú-Paz; 1998)

- Build and sample a Bayes net instead of crossover/mutation.
- Model structure and parameters adapt to problem landscape.
Bayesian network has two parts

- **Structure**
  - Structure is defined by directed acyclic graph (DAG).
  - Nodes define variables (string positions).
  - Edges define direct conditional dependencies.

- **Parameters**
  - Parameters specify conditional probabilities of each variable given its parents (variables that this variable depends on).

```
X0  p(X0)
0   0.40
1   0.60

X1  p(X1 | X0)
0 0 0.10
0 1 0.60
1 0 0.90
1 1 0.40

X2  p(X2 | X0 X1)
0 0 0 0.20
0 0 1 0.55
0 1 0 0.70
0 1 1 0.12
1 0 0 0.80
1 0 1 0.45
1 1 0 0.30
1 1 1 0.88
```
Building a Bayesian Network

Two components to learn

1. Learn structure.
2. Learn parameters.
Learning Structure of a Bayesian network

Greedy algorithm

- Start with an empty network (no edges).
- Add one edge at a time that appears to be the best.
- Stop when no more improvement possible.

Deciding on best structures

- Use a scoring metric to evaluate competing structures:

\[
BIC(B) = \sum_{i=1}^{n} \left( -H(X_i|\Pi_i)N - 2^{\Pi_i} \frac{\log_2(N)}{2} \right)
\]

- \(H(X_i|\Pi_i)\) is the conditional entropy of \(X_i\) given parents \(\Pi_i\)
- \(n\) is the number of variables
- \(N\) is the population size
Maximum likelihood estimation

- Parse data (population of points).
- Make maximum likelihood estimate of parameters:

\[ p(X_i = x_i | \Pi_i = \pi_i) = \frac{m(X_i = x_i, \Pi_i = \pi_i)}{m(\Pi_i = \pi_i)} \]

- \( m(X_i = x_i, \Pi_i = \pi_i) \): is the number of instances with \( X_i = x_i \) and \( \Pi_i = \pi_i \)
- \( m(\Pi_i = \pi_i) \) denotes the number of instances with \( \Pi_i = \pi_i \)
Compact Genetic Algorithm (cGA)

**cGA (Harik et al., 1998)**

- Use **probability vector** as a model:
  \[ p = (p_1, p_2, \ldots, p_n) \]
  where \( p_i \) denotes probability of 1 in \( i \)th position.

- Probability vector is initialized as
  \[ p = (0.5, 0.5, \ldots, 0.5) \]

- **Population is fully eliminated**—Each iteration generates only few solutions, which are used to update the model.
Compact Genetic Algorithm

Iteration

- Generate 2 solutions from the current vector.
- Run a tournament based on fitness: $w$ is winner, $l$ is loser.
- Pretend as if $w$ replaced $l$ in the population of size $N$.
- Frequencies of 1s change as follows

$$p^*_i = \begin{cases} 
  p_i & \text{if } w_i = l_i \\
  p_i + 1/N & \text{if } w_i > l_i \\
  p_i - 1/N & \text{if } w_i < l_i 
\end{cases}$$

Reasoning

- The vector moves closer to $w$ and further from $l$
  - $w_i = l_i$: 0 replaces 0 or 1 replaces 1 $\Rightarrow$ no change.
  - $w_i > l_i$: 1 replaces 0 $\Rightarrow$ increase proportion of 1s by $1/N$.
  - $w_i > l_i$: 0 replaces 1 $\Rightarrow$ decrease proportion of 1s by $1/N$. 

Martin Pelikan, Kumara Sastry, David E. Goldberg

iBOA: The Incremental Bayesian Optimization Algorithm
Tournament result

Winner \( (0, 1, 1, 0, 1, 1, 0) \)

Loser \( (1, 0, 1, 0, 0, 1, 1) \)

Probability vector change for \( N = 100 \)

Old \( (0.66, 0.17, 0.42, 0.37, 0.25, 0.14, 0.78) \)

New \( (0.65, 0.18, 0.42, 0.37, 0.26, 0.14, 0.77) \)
cGA on 100-bit Onemax

Onemax: \( f(X_1, \ldots, X_n) = \sum_{i=1}^{n} X_i \)
Why iBOA?

**cGA**
- cGA does not need a population.
- But cGA is limited to simple problems.
- Similar with other incremental EDAs...

**BOA**
- BOA can encode contexts of arbitrary order.
- This allows BOA to solve much more complex problems.
- But BOA needs an explicit population of points.
- Similar with other multivariate EDAs...

**iBOA = BOA + cGA**
- How can we combine the benefits of the two?
What needs to be done?

- Update structure incrementally.
- Update parameters (conditional probabilities) incrementally.
Basic setup

- Consider marginal probability $p(X_i, \Pi_i)$.
- Conditional probabilities can be computed as

$$p(X_i|\Pi_i) = \frac{p(X_i, \Pi_i)}{p(\Pi_i)} = \frac{p(X_i, \Pi_i)}{\sum_{x_i} p(X_i = x_i, \Pi_i)}$$

Initialization of $p(X_i, \Pi_i)$ according to uniform distribution

$$p(X_i = x_i, \Pi_i = \pi_i) = \frac{1}{2^{|\Pi_i|} + 1}$$

Update the marginal probability table for $P(X_i, \Pi_i)$

- Increment probability of entry consistent with $w$ by $1/N$.
- Decrement probability of entry consistent with $l$ by $1/N$. 
Incremental Parameter Updates: Example for $N = 100$

**Tournament result**

- **Winner**: $(0, 1, 1, 0, 1, 1, 0)$
- **Loser**: $(1, 0, 1, 0, 0, 1, 1)$

**Marginal probability table update**

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_4$</th>
<th>$X_5$</th>
<th>$p_{old}(X_1, X_4, X_5)$</th>
<th>$p_{new}(X_1, X_4, X_5)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td><strong>1</strong></td>
<td><strong>0.05</strong></td>
<td><strong>0.06</strong></td>
</tr>
<tr>
<td>0</td>
<td><strong>1</strong></td>
<td>0</td>
<td>0.12</td>
<td>0.12</td>
</tr>
<tr>
<td>0</td>
<td><strong>1</strong></td>
<td><strong>1</strong></td>
<td><strong>0.10</strong></td>
<td><strong>0.10</strong></td>
</tr>
<tr>
<td><strong>1</strong></td>
<td><strong>0</strong></td>
<td>0</td>
<td><strong>0.26</strong></td>
<td><strong>0.25</strong></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td><strong>1</strong></td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>1</td>
<td><strong>1</strong></td>
<td>0</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td><strong>1</strong></td>
<td><strong>1</strong></td>
<td><strong>1</strong></td>
<td><strong>0.14</strong></td>
<td><strong>0.14</strong></td>
</tr>
</tbody>
</table>
Incremental structure updates

- Want to be able to change structure over time.
- Can start with empty network and build up over time.
- But we can’t predict how the structures will look in future!
- So we have to consider all possibilities.

Efficiency becomes a problem

- Consider networks of at most $k$ parents.
- Must maintain all marginal probabilities of order $k + 1$.
- This leads to $\binom{n}{k+1}$ tables or $\Omega(2^k n^{k+1})$ probabilities!

Challenge

- Can we do better?
### Challenge: Some numbers

#### Illustration

<table>
<thead>
<tr>
<th>$n$</th>
<th>$k$</th>
<th>$\binom{n}{k+1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>210</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>38,760</td>
</tr>
<tr>
<td>30</td>
<td>5</td>
<td>593,775</td>
</tr>
<tr>
<td>40</td>
<td>5</td>
<td>3,838,380</td>
</tr>
<tr>
<td>50</td>
<td>5</td>
<td>15,890,700</td>
</tr>
</tbody>
</table>

Martin Pelikan, Kumara Sastry, David E. Goldberg

iBOA: The Incremental Bayesian Optimization Algorithm
One step at a time

- Start with empty network.
- Always store enough probabilities to add one more edge.
- Parameters are updated in each iteration.
- Once new addition seem worthy (scoring metric), do it.
  - Add new edge.
  - Add new parameters, needed for yet another addition.
  - New probabilities created from the old ones using independence assumptions to fill in the holes.
- Need only $O(n^2)$ probability tables to maintain.
Challenge Met: Some numbers

Illustration

<table>
<thead>
<tr>
<th>$n$</th>
<th>$k$</th>
<th>$\binom{n}{k+1}$</th>
<th>$n^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>210</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>38,760</td>
<td>400</td>
</tr>
<tr>
<td>30</td>
<td>5</td>
<td>593,775</td>
<td>900</td>
</tr>
<tr>
<td>40</td>
<td>5</td>
<td>3,838,380</td>
<td>1,600</td>
</tr>
<tr>
<td>50</td>
<td>5</td>
<td>15,890,700</td>
<td>2,500</td>
</tr>
</tbody>
</table>
How does iBOA work?

- Start with an empty network (similar to cGA).
- Grow the network over time.
- Update both parameters and structure incrementally.
- Population is not needed at all!
Experimental Setup

Description of experiments

- Use bisection to find adequate population size (within 5%).
- Require 30 successful runs (out of 30 runs).
- Upper bound the number of generations by $n$.
- Consider multiple problem sizes.
- Analyze scalability (num. evaluations w.r.t. problem size).
- Use tournaments of size 4 (best and worst update model).

Test problems

- Trap-4.
- Trap-5.
Test Problems

Trap-4: Concatenated trap of order 4

\[ f_{\text{trap}4} = \frac{n}{4} \sum_{i=1}^{n/4} \text{trap}_4(X_{4i-3} + X_{4i-2} + X_{4i-1} + X_{4i}) \]

Trap-5: Concatenated trap of order 5

\[ f_{\text{trap}5} = \frac{n}{5} \sum_{i=1}^{n/5} \text{trap}_5(X_{5i-4} + X_{5i-3} + X_{5i-2} + X_{5i-1} + X_{5i}) \]

Trap of order \( k \)

\[ \text{trap}_k(u) = \begin{cases} \\ k & \text{if } u = k \\ k - u - 1 & \text{otherwise} \end{cases} \]
Results on Trap-4

iBOA: The Incremental Bayesian Optimization Algorithm

Martin Pelikan, Kumara Sastry, David E. Goldberg
Future Work

Diversity maintenance and elitism

- How to incorporate elitism?
- How to incorporate niching for useful diversity maintenance?

Memory efficiency

- Model still takes considerable chunk of memory.
- Asymptotic growth of memory complexity is the same for BOA & iBOA.
- How can we create more efficient representations?
  - Use local structures in Bayesian networks.
  - Use alternative model-building and sampling procedures.

Model building

- Add more operators (e.g. edge removal, edge reversal).
Summary

- Proposed incremental Bayesian optimization alg. (iBOA).
- Analyzed iBOA performance.

Conclusions

- Design of incremental EDAs with multivariate models possible.
- Opened the door to new generation of incremental EDAs.
- Incremental EDAs with multivariate models can scalably solve decomposable problems without maintaining populations of candidate solutions.
- But some challenges still remain
  - Memory cost is still considerable.
  - Not so easy to incorporate niching and elitism.
Acknowledgments

- NSF; NSF CAREER grant ECS-0547013.
- U.S. Air Force, AFOSR; FA9550-06-1-0096.
- University of Missouri; High Performance Computing Collaboratory sponsored by Information Technology Services; Research Award; Research Board.