BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Information Technology

Department of Computer Systems

Ing. Jiří Očenášek

PARALLEL ESTIMATION OF DISTRIBUTION ALGORITHMS

PARALELNÍ EVOLUČNÍ ALGORITMY VYUŽÍVAJÍCÍ PRAVDĚPODOBNOSTNÍ MODELY

SHORT VERSION OF PHD THESIS

Study field:	Information technology
Supervisor:	Ing. Josef Schwarz, CSc.
Opponents:	Prof. Ing. Vladimír Kvasnička, DrSc.
	Doc. Ing. Pavel Ošmera, CSc.
	Ing. Aleš Gottvald, CSc.
Presentation date:	January 28, 2003

KEY WORDS

Bayesian network, Gaussian network, Estimation of Distribution Algorithm, Factorized Distribution Algorithm, Bayesian Optimization Algorithm, Univariate Marginal Distribution Algorithm, Bivariate Marginal Distribution Algorithm, Bayesian-Dirichlet metric, scoring metric, dependency structure, machine learning, linkage learning, decision graph, kernel distribution, mixture model, formal description, building block corruption, convergence, hypergraph bisectioning, knapsack problem, additively decomposable function, multiobjective optimization, Pareto set, niching, Pareto-strength fitness, epsilon dominance, pipelined processing, distributed processing, multithreaded processing, scalability, EDA, BOA

KLÍČOVÁ SLOVA

genetický algoritmus, Bayesovský optimalizační algoritmus, rozhodovací stromy, rozhodovací grafy, cenová funkce, matoucí funkce, dekompozice, členění grafů, knapsack problém, evoluce, optimalizace, Bayesovská síť, Gaussovská síť, Paretovská množina, epsilon dominance, EDA, BOA

MÍSTO ULOŽENÍ

Knihovna Fakulty Informačních Technologií VUT v Brně, Božetěchova 2, 612 66 Brno

ISBN 80-214-2329-3
ISSN 1213-4198

Contents

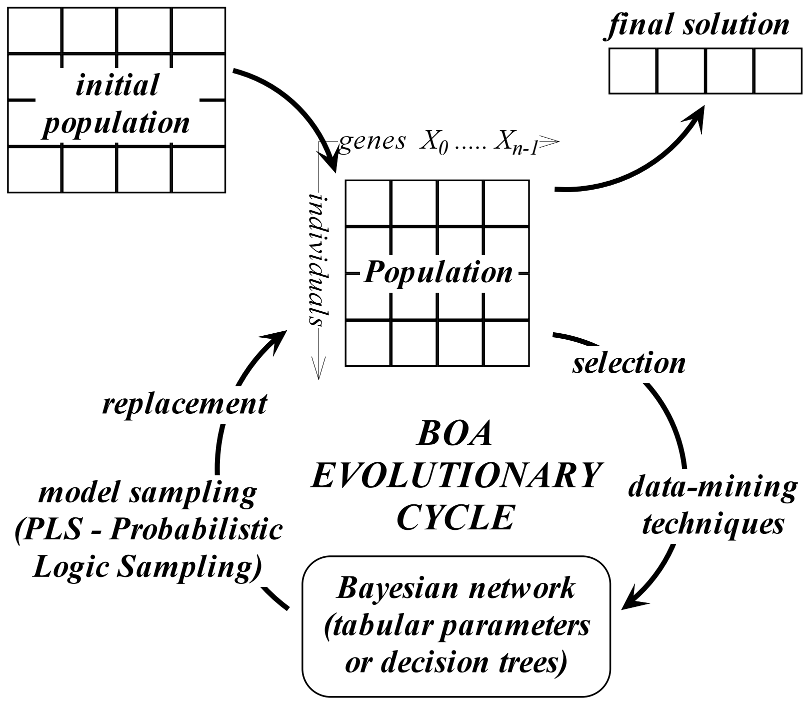
1 INTRODUCTION

The general procedure of Estimation Distribution Algorithm (EDA) is similar to that of GA, but the classical recombination operators are replaced by probability estimation followed by probability sampling. First, the initial population of EDA is generated randomly. In each iteration, promising solutions are selected from the current population of candidate solutions and the true probability distribution of these selected solutions is estimated. New candidate solutions are then generated by sampling the estimated probability distribution. The new solutions are then incorporated into the original population, replacing some of the old ones or all of them. The process is repeated until the termination criteria are met.

The Bayesian Optimization Algorithm (BOA) [6] attracted much attention during few last years. It uses Bayesian network (BN) to factorize the structure of solved problem:

$$p(X_0,...,X_{n-1}) = \prod_{i=0}^{n-1} p(X_i \mid \boldsymbol{\Pi}_i), \tag{1}$$

where Π_i denotes the set of genes (random variables) that determine the value of gene X_i in the dependency graph of Bayesian network. The methods and metric for BN building were adopted from the area of data mining, namely from Heckerman, Geiger & Chickering [4]. The Bayessian Dirichlet metric (BDe) is used to measure the quality of the network. The complete BOA evolutionary cycle is shown in the following figure.



initial population
final solution
genes X_0 ..... X_{n-1}
individuals
Population
selection
replacement
BOA EVOLUTIONARY CYCLE
model sampling (PLS - Probabilistic Logic Sampling)
data-mining techniques
Bayesian network (tabular parameters or decision trees)

Fig. 1.1: BOA evolutionary cycle.

2 RESEARCH OBJECTIVES

2.1 Open problems

There is currently a considerable amount of research surrounding Estimation of Distribution Algorithms. However, the following open problems can be identified:

1. There does not exist a complete formal framework for EDAs.
2. The research on general models for continuous domains should be refined – the present models are computationally expensive or too simplified.
3. Computational complexity of model construction significantly restricts the size of solvable problems.
4. Existing EDAs are not applicable to problems with multiple objectives.
5. Modular development system for rapid prototyping of EDA applications is missing.
6. The efficiency of EDAs for real world problems was not investigated – most empirical studies deal only with synthetic benchmarks with bounded degree of epistasis.

2.2 Research outline

This dissertation solves all the open problems stated in previous section. The main goals include :

1. Identification of the common features of existing EDAs and creation of their formal description.
2. Development of new probabilistic model which is effectively applicable to real-valued or even mixed optimization problems.
3. Time analysis of main EDA components for the purpose of algorithmic simplification and parallel implementation. Development of new parallel methods and heuristics providing for realtime performance.
4. Extension of EDA framework for multiobjective optimization.
5. Design and implementation of modular development system for rapid prototyping of EDA applications.
6. Investigation of EDA performance on real world problems, namely from the area of circuit design.

3 APPROACH

3.1 Formal description of EDA algorithms

The following paragraph describes an original formal definition of Estimation of Distribution Algorithm. The reasons for formal approaches are straightforward since: (i) problem specification is rigorous, (ii) a mathematical apparatus can be applied to investigate their properties, and (iii) tools for automatic analysis, verification and design can be developed. As the baseline I use Back's formal definition of evolutionary algorithms presented in [1].

Definition 3.1 (Estimation of Distribution Algorithm) :
An Estimation of Distribution Algorithm is defined as an 11-tuple

$$\text{EDA} = \left(I, \Phi, \mathcal{M}, \Psi, \rho, \omega, s, r, \iota, \mu, \lambda\right), \tag{3.1}$$

where:

(i) $I = A^n$ is the space of individuals of length n over the domain A,

(ii) $\Phi : I \to \mathbb{R}$ denotes a fitness function assigning real values to individuals (in EDA framework often denoted as $F(\boldsymbol{x})$),

(iii) $\mathcal{M} = \{\mathrm{M} \mid \mathrm{M} : I \to [0,1]\}$ is the space of admissible probabilistic models,

(iv) $\Psi : I^{\mu+\lambda} \to I^{\mu+\lambda}$ is the generation transition function,

(v) $\rho : I^{\mu} \times \mathcal{M} \to \mathbb{R}$ denotes the metrics for evaluating probabilistic models,

(vi) $\omega : \mathcal{M} \to I^{\lambda}$ denotes the model sampling operator (non-deterministic),

(vii) $s : I^{\mu+\lambda} \to I^{\mu}$ is the $(\mu+\lambda)$-selection operator, which selects the parent population $D(\mathrm{t})$ from population $P(\mathrm{t})$. The most frequent truncation selection operator is defined as $s_{(\mu+\lambda)}(P(\mathrm{t})) = D(\mathrm{t})$ such that: $\exists \boldsymbol{x} \in P(\mathrm{t}) \setminus D(\mathrm{t}) : (\exists \boldsymbol{x}' \in D(\mathrm{t}) : \Phi(\boldsymbol{x}) > \Phi(\boldsymbol{x}'))$,

(viii) $r : I^{\mu+\lambda+\lambda} \to I^{\mu+\lambda}$ is the replacement operator, which creates new population $P(\mathrm{t+1})$ from the old population $P(\mathrm{t})$ and offspring population $O(\mathrm{t})$. The most frequent replace-worst operator is defined as $P(\mathrm{t+1}) = r_{(\mu+\lambda+\lambda)}(P(\mathrm{t}), O(\mathrm{t})) = s_{(\mu+\lambda)}(P(\mathrm{t})) \cup O(\mathrm{t})$,

(ix) $\iota : I^{\mu+\lambda} \to \{true, false\}$ is the termination criterion,

(x) μ is the number of parent individuals (often denoted as N), while λ denotes the number of offspring individuals (usually equal to N, so $L = \lambda + \mu = 2 * N$).

This definition is based on high-level concept, where each probabilistic model *M* is a mapping from the space of strings to the space of probability values. This is captured in the generation transition function

$$\Psi(P(t)) = r\left(P(t), \omega^{\lambda}\left(\arg\max_{M \in \mathcal{M}} \rho(s(P(t)), M)\right)\right) \tag{3.2}$$

Iterated application of transition function generates a population sequence and leads to definitions of the running time and the result of EDA.

3.1 Solving continuous & mixed problems

The most straightforward way how to solve continuous benchmarks by BOA is to transform each continuous parameter into binary parameter and use the Bayesian network model for such a binary encoding. See the papers [9] and [11] for further details on this approach. Unfortunately, this approach is not scalable. With the increasing precision of encoding the BN parameter k becomes unmanageably large and the complexity of BN construction grows exponentially.

The other way of solving continuous optimization problem by EDA is to let the parameters Y_i be continuous (like in evolutionary strategies) and find a proper model for this continuous domain. For continuous domains there is an IDEA approach [2] used to discover and encode the parameter dependencies – estimating Gaussian networks, Gaussian kernels and mixtures of Gaussians. However, these models are not compatible with discrete domains.

This is the reason for proposing new EDA model based on Classification And Regression Trees (CART), which is usable also for continuous and categorical domains:

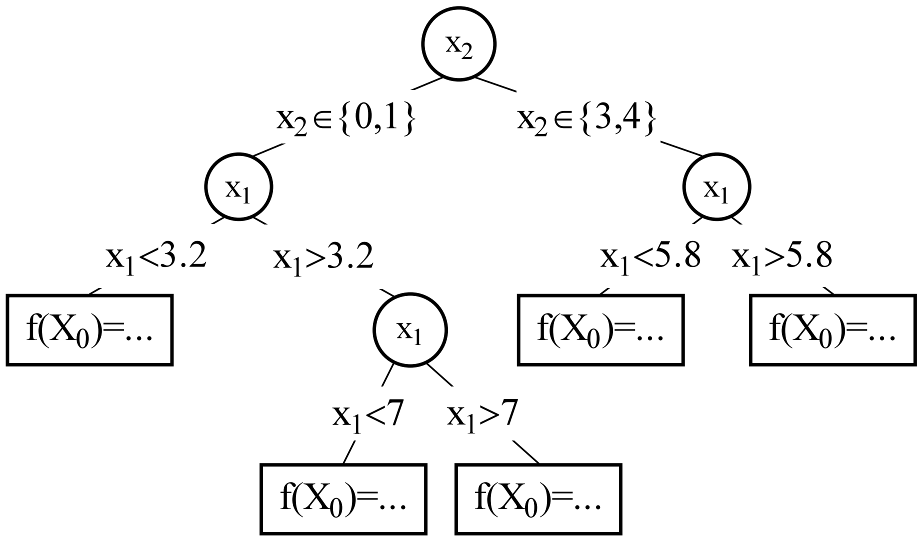


Fig. 3.1: An example of CART model which comprises of one categorical split on variable X_2 and three continuous splits on variable X_1. The leaf nodes conditionally determine the probability density functions for target variable X_0.

3.2 Parallel processing

The most significant difference between classical GA and the EDA algorithm lies in the character of creation of new population. In case of classical GA the crossover is pairwise operation and can be performed even locally, but in case of EDA the classical models of parallelism are useless, because the accuracy of estimated model decreases rapidly with decreasing size of parent population. The only way how to design a parallel EDA algorithm is to propose new paradigms for parallel construction and sampling of probabilistic model in the pseudo-sequential manner. The goal is to utilize more processors when searching for a good model.

Empirical analysis shows that nearly all the execution time of sequential BOA is spent to find the dependence graph of Bayesian network. Thus, some method for parallel BN construction needs to be proposed. Our consolation is that the BDe metric is separable and can be written as a product of n factors, where i-th factor expresses the influence of edges ending in the variable X_i. Thus, for each variable X_i the set of parent variables Π_i can be determined independently. It is possible to utilize up to n processors, each processor corresponds to one variable X_i and it examines only edges leading to this variable.

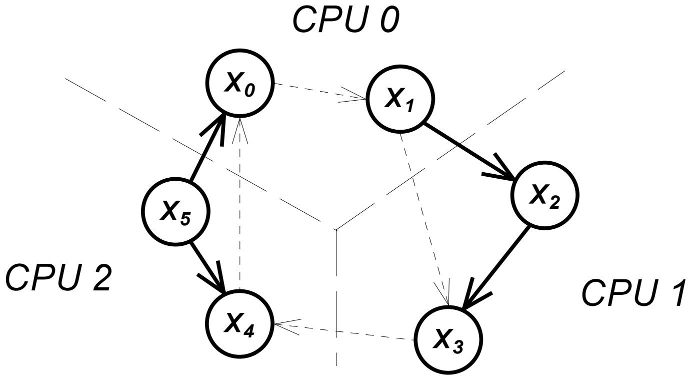


Fig. 3.2: CPU_0 is adding edges ending in nodes 0 and 1; CPU_1 is adding edges ending in nodes 2 and 3; CPU_2 is adding edges ending in nodes 3 and 4.

In Fig. 3.2 you see that the naïve parallel BN construction might produce the unwanted cycles (dashed lines). The addition of edges is parallel, so we need an additional mechanism to keep the dependence graph acyclic. The most advantageous way how to handle this problem is to predetermine the topological ordering of nodes in advance. At the beginning of each generation, the random permutation of numbers $\{0,1,\ldots,n\text{-}1\}$ is created and stored in the $\boldsymbol{o}=(o_0,o_1,\ldots,o_{n-1})$ array. Each processor uses the same permutation. The direction of all edges in the network should be consistent with the ordering, so the addition of an edge from X_{o_j} to X_{o_i} is allowed if only $j<i$. Evidently, the variable X_{o_0} has no predecessor and is forced to be independent, thus the space of possible networks is reduced. To compensate this phenomenon the processors generate new permutation after each generation.

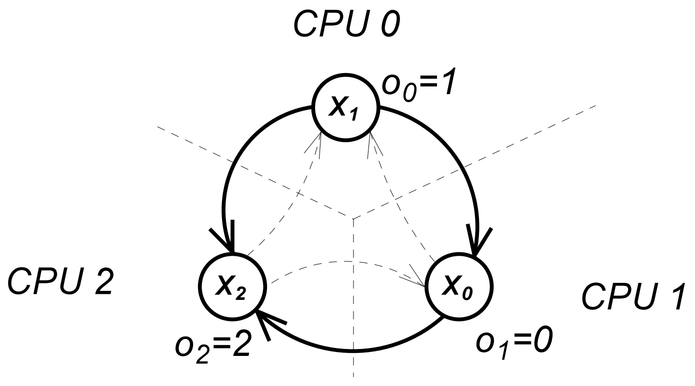


Fig. 3.3: Example of BN construction with 3 processors and n=3. Permutation vector ***o****=(1,0,2) determines the topological ordering of nodes such that the dashed edges are not allowed.*

3.3 Multiobjective optimization

Many real-world problems have multiple non-commensurable and often competing objectives. The main attention is focused on the incorporation of well known multiobjective niching techniques into structure of EDA to find the so called Pareto optimal set. I have designed two variants of multiobjective BOA according to Pareto-strength concept and epsilon-dominance concept.

3.3.1 Pareto-strength concept

The standard BOA is able to find mostly one optimal solution at the end of the optimization process, when the whole population is saturated by phenotype-identical individuals. To overcome this problem a promising Pareto-strength niching technique is applied, published in [12]. The space of objective vectors is divided into rectangles corresponding to the combinations of solutions from Pareto-set. The Pareto-strength fitness assignment tries to replace the objective vector by the scalar fitness value according the following two rules (see Fig. 3.4):

1. *Individuals dominated by smaller number of nondominated individuals are preferred.*
2. *Dominated individuals having more neighbours in their „niche“ are penalized.*

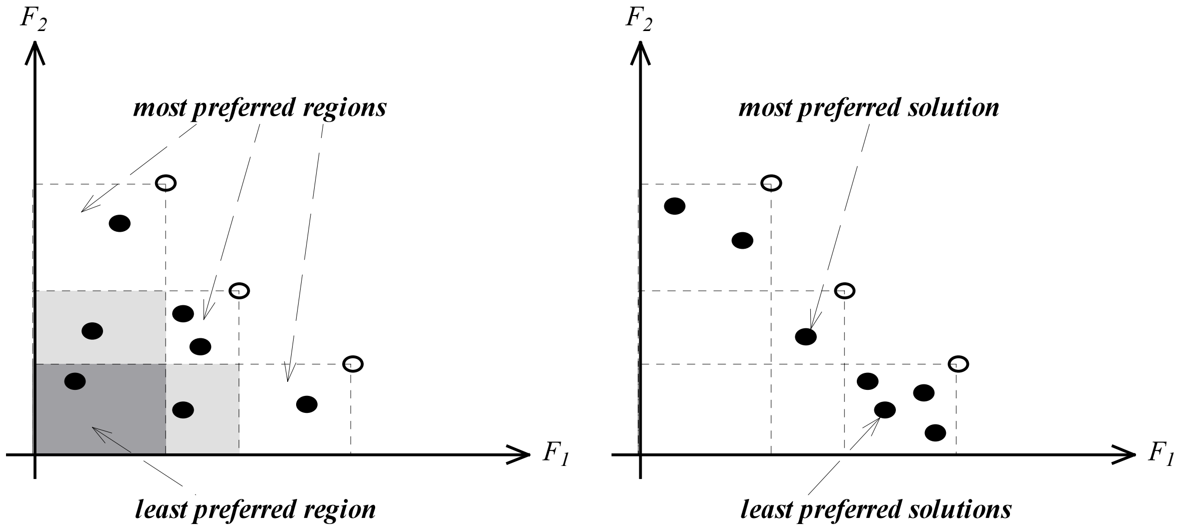


Fig. 3.4: Illustration of intuitive rules for Pareto-strength fitness assignment

3.3.2 Epsilon-dominance concept

The utilization of epsilon-dominance concept is a result of joint research with Marco Laumanns from ETH Zűrich, one of the authors of well known SPEA2 algorithm. We derived design guidelines for a successful selection scheme in EDAs:

1. *Elitism (to preclude the problem of gradual worsening and enable convergence to the Pareto set),*
2. *Diversity maintenance in objective space (to enable a good approximation of the whole Pareto set), and*
3. *Diversity maintenance in decision space (to avoid redundancy and provide enough information to build a probabilistic model).*

In connection with the construction of mixed decision trees probabilistic model we developed new selection operator based on ϵ-archives [6]:

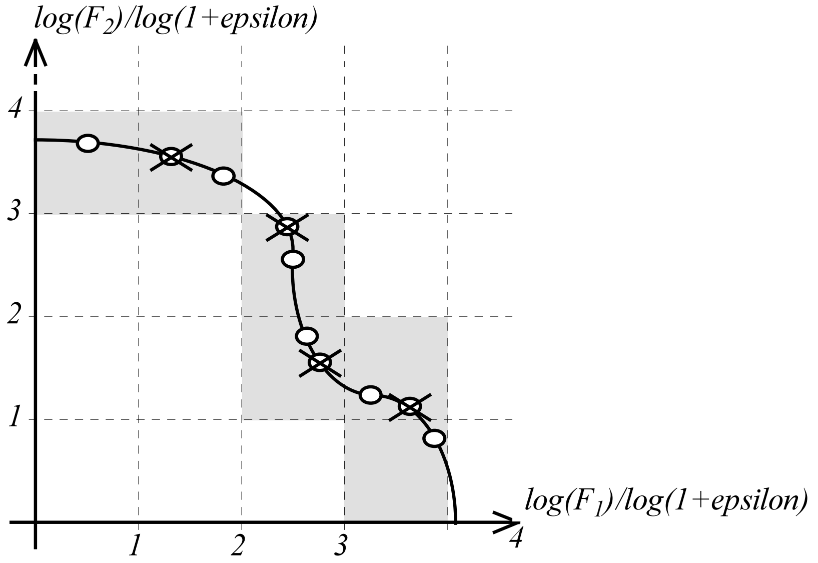


Fig. 3.4: The basic idea of ε-dominance. The objective space is divided into ε-boxes. Each ε-box can contain only one solution. This avoids outbreak of "easy to get" solutions from the middle region.

3.4 Prototyping of BOA applications

The creation of a well-performing classical evolutionary algorithms highly depends on problem encoding, genetic operators and control parameters. In the field of classical EAs there are many development environments for fast prototyping, but in the field of EDAs these tools are missing. The work on DEBOA system for the rapid prototyping of BOA-based applications is an extension of Radim Kostka‘s master thesis [5] I supervised. This system includes all advantageous features of existing development systems:

3.4.1 Visualization

By the visualization I mean the ability of EA development system to provide as many information about the evolution as possible. The common visualization can be divided into two classes:

- visualization of EA evolution process
- visualization of current population distribution in solution space

3.4.2 Extensibility

Modern EA toolkits should be implemented in a modular way, such that a new operator, visualization or fitness function can be added. After building a final source code for solved optimization task two extensibility modes can be used:

- Interpreted code
- Binary code using DLL files

The interpreted mode uses a built-in interpreter of pseudo-programming language in which the user code can be written. The binary one is faster, because the precompiled binary code is dynamically loaded and executed (for example from .DLL file or Java .class file).

3.4.3 Reusability

Most of EA toolkits support only prototyping, but some of them can also export settings and operators into final application such that the optimization engine needs not to be programmed again. For example the core of reusable EA toolkit is contained in .DLL file and its API functions are called from the source code generated according to optimal settings found.

3.4.4 Portability

By the portability I mean the transfer of EA toolkit from one platform to another. Usually the commercial toolkits contain platform-specific binaries whereas the open-source toolkits are easily portable.

4 THE RESULTS

4.1 MBOA

In Mixed Bayesian Optimization Algorithm (MBOA) design I have focused on solving problems with mixed continuous-discrete parameters. For each “target” random variable X_i MBOA builds one decision tree. The parent variables Π_i contained in split nodes of *i*-th decision tree are used to cut the space of all chromozomes into parts, where the variable X_i is more linear or predictable.

The building of decision trees starts from empty trees and it recursively adds the splitting nodes until no splitting is favourable:

```
Function RecursiveSplitting(Population Pop, TargetVariable $X_i$,
         ListOfCandidateSplitVariables Pa) : DecisionTreeNode
Begin
  $f_{Temp}$ := EstimateElementaryPDF(Pop,$X_i$,Pa);
  If "$f_{Temp}$ is sufficiently detailed" then return new
LeafNode($f_{Temp}$);
  For Each Variable $X_j$ in Pa do
     $E_j$ := Find_optimal_threshold_of_$X_j$_with_respect_to_$X_i$;
     ModelGain := Evaluate_the_split_quality("$X_j \le E_j$", $X_i$);
     Save the $X_j$ and $E_j$ with the highest ModelGain;
  End for
  Pop1 := SelectIndividuals (Pop,"$X_j \le E_j$");
  Pop2 := SelectIndividuals (Pop,"$X_j > E_j$")
  return new SplitNode(new SplitCondition("$X_j \le E_j$"),
                 RecursiveSplitting(Pop1,$X_i$, Pa\{$X_j$}),
                 RecursiveSplitting(Pop2,$X_i$, Pa\{$X_j$}));
End;
```

You see that the continuous-valued decision attributes are incorporated into the learned tree by dynamically defining new binary attributes that partition the continuous attribute value into two intervals ($Xj \le Ej$ and $Xj > Ej$). The threshold Ej is selected among the candidate thresholds to maximize the metrics. The step of split condition finding and evaluation is essential. The algorithm uses a greedy search, that is, it picks the best candidate and never looks back to reconsider earlier choices. The choice is based on the equation we derived from the Bayesian-Dirichlet metrics:

$$gain(X_i,X_j)=\frac{\Gamma\left(\sum_{x_j}\sum_{x_i}(m(x_i,x_j)+1)\right).\prod_{x_j}\prod_{x_i}\Gamma\left(m(x_i,x_j)+1\right)}{\left(\prod_{x_j}\Gamma\left(\sum_{x_i}(m(x_i,x_j)+1)\right)\right).\left(\prod_{x_i}\Gamma\left(\sum_{x_j}(m(x_i,x_j)+1)\right)\right)} \tag{4.1}$$

where X_i is the variable for which the tree is being constructed, X_j is the parent variable - possible split, and $m(x_i,x_j)$ is the number of individuals having $X_i=x_i$ and $X_j=x_j$. Note that the splitting is done recursively, so $m(x_i,x_j)$ is determined only from the subpopulation being splitted. We often use the logarithm of this metric, which avoids multiplication operations.

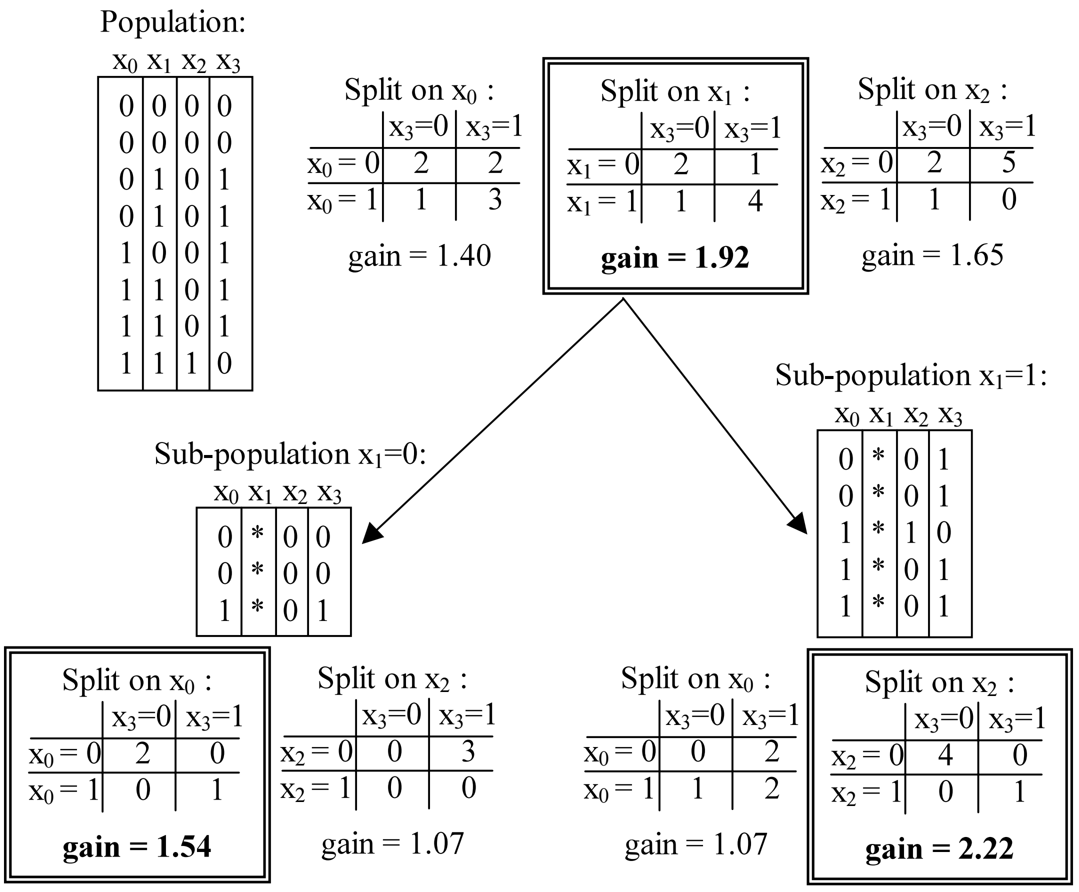
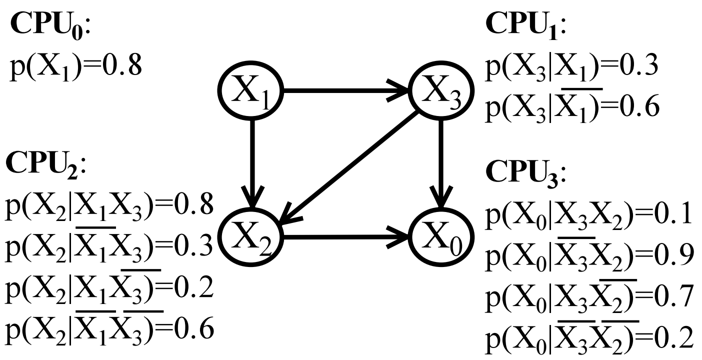


Fig. 4.1: Example of recursive building of decision tree for x_3.

4.2 PBOA

The pipelined algorithm PBOA is the first of my three parallel BOA algorithms. All these algorithms use parallel dependency graph construction based on the principle of restricted ordering of nodes, they differ mainly in the way how the offspring is being generated. PBOA was proposed for fine-grained type of parallelism. The target platform for its implementation can be for example the Transputers with their tightly-connected communication channels, or one-purpose MIMD processor. Each processing element has its own local memory with complete copy of the whole population.

Let us consider usage of m=n processors (resp. processing elements). First the BN is constructed. The i–th processor determines the set of parent variables $\boldsymbol{\Pi}_{o_i}$ and it estimates the parameters of CPD (Conditional Probability Distribution) for $p(X_{o_i} \mid \boldsymbol{\Pi}_{o_i})$. Fig. 4.2 shows an example of parallel construction of BN with n=4 variables and m=4 CPUs, including the parallel estimation of local CPDs.



CPU$_0$:
$p(X_1)=0.8$
CPU$_1$:
$p(X_3|X_1)=0.3$
$p(X_3|\overline{X_1})=0.6$
CPU$_2$:
$p(X_2|X_1X_3)=0.8$
$p(X_2|\overline{X_1}X_3)=0.3$
$p(X_2|X_1\overline{X_3})=0.2$
$p(X_2|\overline{X_1}\overline{X_3})=0.6$
CPU$_3$:
$p(X_0|X_3X_2)=0.1$
$p(X_0|\overline{X_3}X_2)=0.9$
$p(X_0|X_3\overline{X_2})=0.7$
$p(X_0|\overline{X_3}\overline{X_2})=0.2$
X_1
X_3
X_2
X_0

Fig. 4.2: An example - splitting of BN construction between 4 processing elements. The ordering permutation is ***o****=(1,3,2,0), so the variable X_1 is independent. Parts of BN (including local CPD tables) which are estimated by different CPUs have different color shades.*

Now, let us focus on the pipelined offspring generation. PBOA can generate offspring in a linear pipeline way, because in the fine-grained architecture there are negligible communication latencies. It takes *n* cycles to generate the whole chromozome. For example let us consider the first chromozome. Its o_0-th bit is generated independently in processor number 0 at time *t*. Then, its o_1-th bit is generated in processor number 1 at time *t*+1 conditionally on the o_0-th bit received from neighbouring processor number 0, etc. Generally, X_{o_i} is generated in CPU number *i* at time *t+i* conditionally on $\boldsymbol{\Pi}_{\boldsymbol{o}_i} \subseteq \{X_{o_0},...,X_{o_{i-1}}\}$. The advantage is that each CPD is sampled locally at the place where it had been estimated:

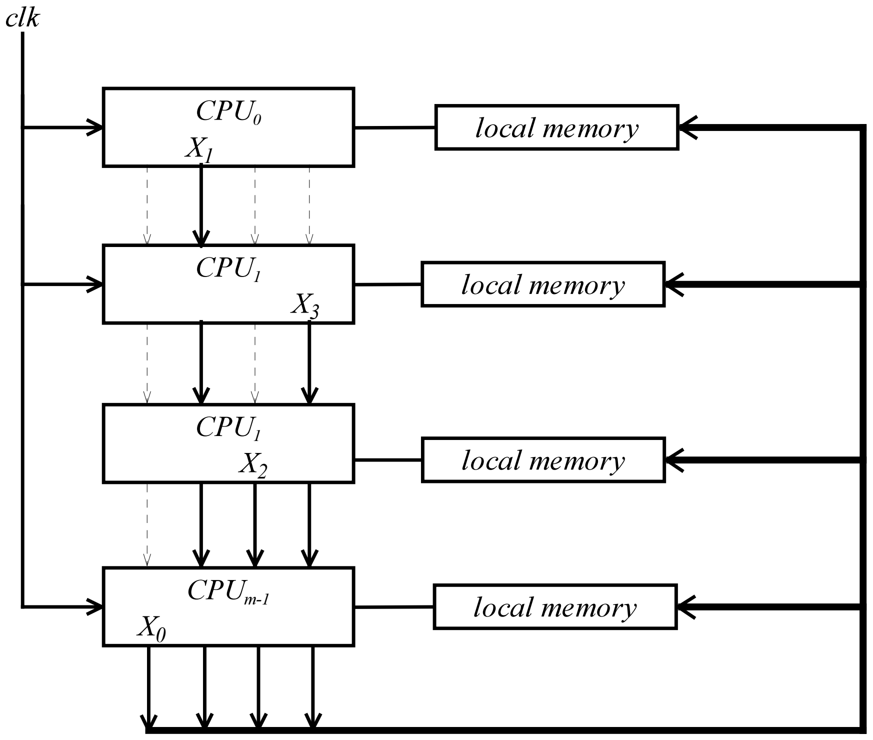


Fig. 4.3: Pipelined generation of ofspring according to the BN constructed in Fig. 4.2. The dashed arrows mean ungenerated variables.

4.3 DBOA

My Distributed Bayesian optimization algorithm (DBOA) was proposed and implemented for coarse-grained type of parallelism. Its target platform is for example the cluster of workstations connected by fast Ethernet network. The experiments were run on the uniform cluster of Sun Ultra 5 workstations with UltraSPARC II processors at 270 MHz. The hardware multiport router has been Summit48 (from Extreme Networks), which is able to transfer up to 100 Mb/s between pairs of distinct workstations simultaneously. I used the MPI C++ library version 1.2 which provides the well known standard for message passing communication.

At the beginning of DBOA cycle each processor has the full copy of the parent population and it constructs the part of BN dependence graph, in the same way like PBOA. Unfortunately, the communication delays are too long to use pipelined generation of offspring. Thus, DBOA uses distributed generation of offspring, each processor generates the whole subpopulation of chromozomes. See Fig. 4.4 for comparison of both types of offspring generation.

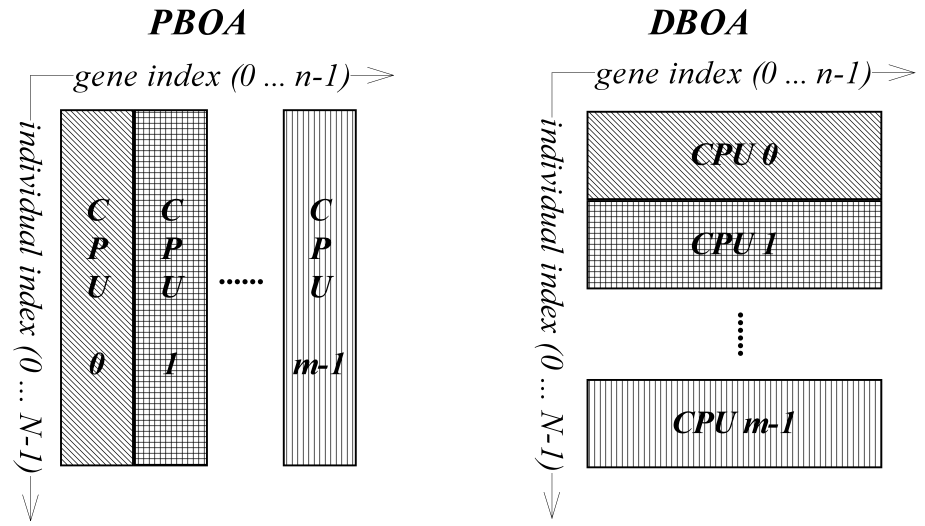


Fig. 4.4: Comparison of offspring generation. PBOA distributes the work between processors „horizontally“ and DBOA „vertically“.

Note that for this kind of offspring generation each processor needs to use the complete probabilistic model, which is constructed piecewise. Thus, a necessary step between model estimation and offspring generation is the gathering of local parts of model. Each processor exchanges its part of model with the other processors. The workload balancing methods ensure that all workstations finish the BN construction and the BN sampling in the same time, because BN gathering or offspring gathering steps can be considered as the barrier synchronization. The complete example of one DBOA timing cycle can be seen in Fig. 4.5.

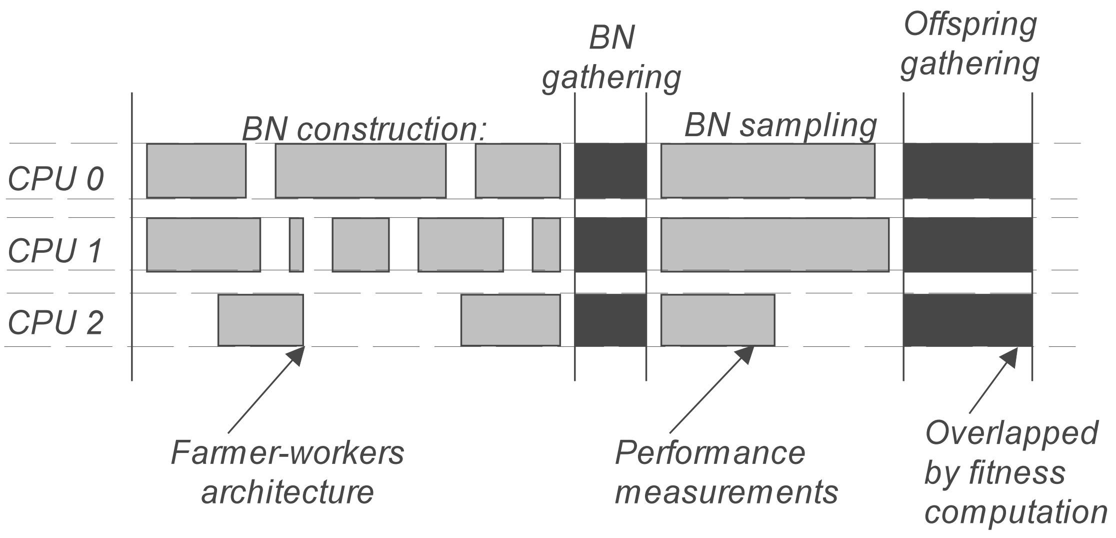


Fig. 4.5: Example of timing diagram for one DBOA cycle.

4.4 Multithreaded MBOA

In MBOA algorithm a set of binary decision trees/graphs is used to express the probability model. Many ideas for parallel MBOA algorithm can be adopted from the DBOA design, namely the concept of restricted ordering of nodes.

The MBOA differs from BOA in the heterogeneous model parameters. It is very difficult task for each process to convert the parameters of decision tree into one linear data stream, which has to be exchanged with the other processes. It would be far more comfortable if each decision tree could be used exactly in the same place where it has been built. Thus, the goal is to design the distributed MBOA algorithm, which does not need to exchange the decision trees between processes. It uses the distributed generation of offspring in the "vertical" manner (see Fig. 4.4 left), but it was designed for use on cluster of workstations. Multithreaded processing was proposed to overcome large communication latencies.

The whole architecture of multithreaded MBOA is shown in Fig. 4.6. The *farmer* thread is responsible for dynamic workload assignment, *builder* threads are responsible for building decision trees and *generator* threads are responsible for generating new genes. The *buffer* threads are used only for buffering dataflow between builders and generators, because Transim does not support buffered channels. All the threads *builder*$_i$, *buffer*$_i$ and *generator*$_i$ are mapped to the same *i*-th workstation.

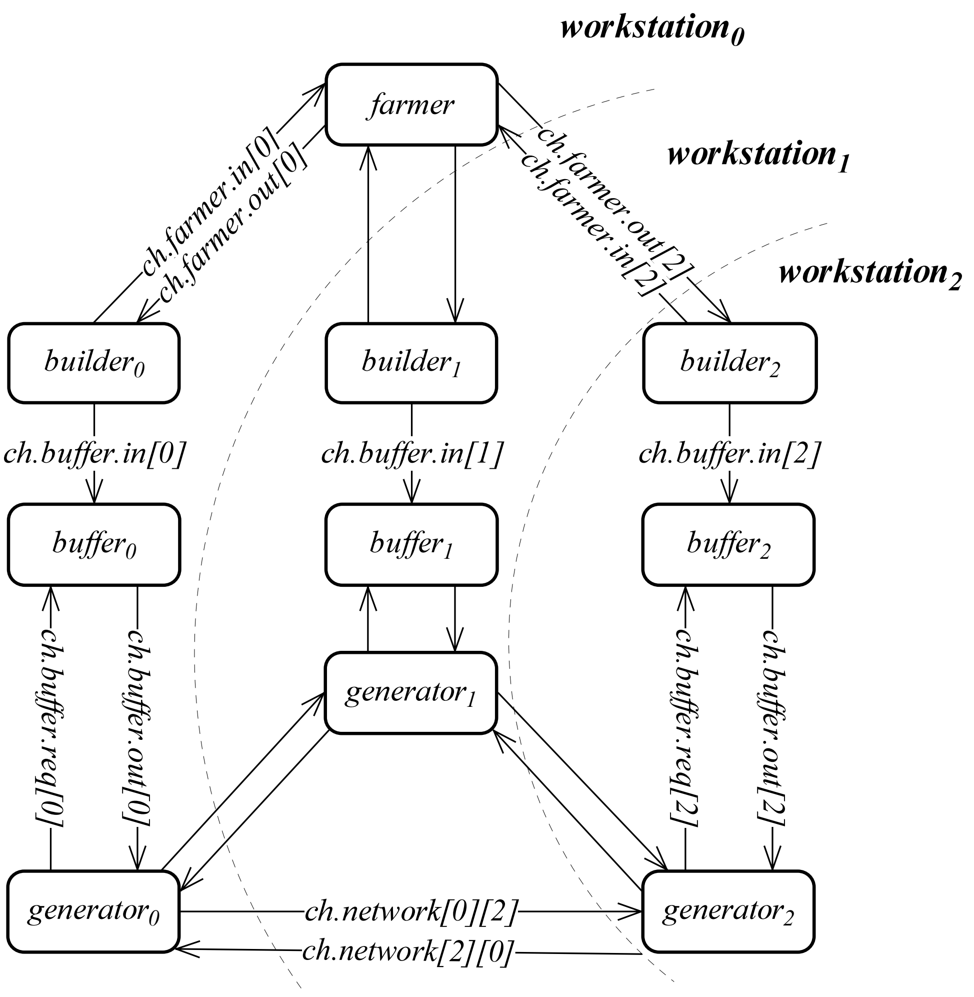


Fig. 4.6: Architecture of multithreaded MBOA in Transim. The dashed curves separate workstations

The multithreaded MBOA algorithm was simulated using the well known TRANSIM tool.

4.5 Pareto BOA

Pareto BOA algorithm is a modification of original Pelikan's BOA code [8] where a Pareto-strength niching technique is applied. The most important part of the Pareto BOA algorithm is the procedure for detection of nondominated solutions (current Pareto front) and sophisticated Pareto-strength fitness calculation. The whole structure of Pareto-strength based BOA algorithm is stated in Fig. 4.7.

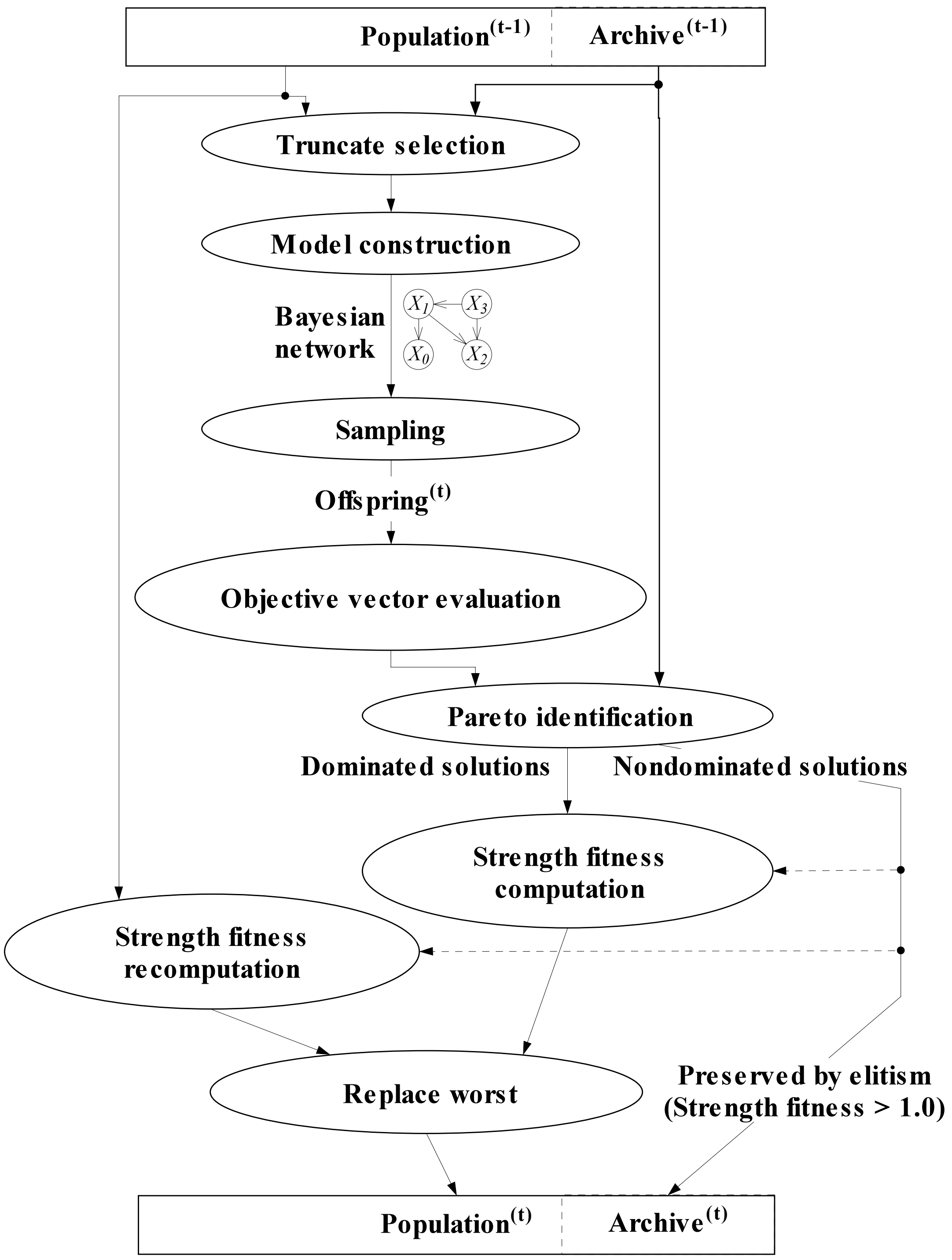


Fig. 4.7: Architecture of Pareto-strength BOA

4.6 (μ+λ,ε) – BMOA

The Pareto-strength BOA algorithm was comparable to the best multiobjective-algorithms of its time - SPEA [12] and NSGA [10]. However, I decided to create new algorithm to be competitive to the recent SPEA2 [13] and NSGA-II [3] algorithms. The aim was also to develop completely new multiobjective technique for MBOA core based on mixed decision trees model. I started the joint research with Marco Laumanns from ETH Zűrich, one of the SPEA2 authors. Our new Bayesian Multi-objective Optimization Algorithm (BMOA) reflects all the new studies related to the theoretical convergence analysis. It has both properties of converging to the true Pareto-optimal front and maintaining a widely spread distribution of solutions.

The ε-archive was proposed for use in classical MOEAs, because it maintains a minimal set of solutions that ε-dominate all other solutions generated so far. However, as this set can become very small, the scheme has to be modified to provide enough decision space diversity for BOA. The new selection operator is described in Alg. 1. The idea is that now also dominated individuals are allowed to survive, depending on the number of individuals that they are dominated by.

```
Algorithm 1 Select(Ar, P, μ, ε)
  Input: old parent set Ar, candidate set P,
                              minimum size μ, approximation factor ε
  Output: new parent set Ar'
  for all x ∈ P do
    B := {y∈Ar | ∀Fi: ⌊log Fi(y) / log (1+ε)⌋ = ⌊log Fi(x) / log (1+ε)⌋ }
    if B = ∅ then
      Ar:=Ar∪{x}
    else if ∃y∈B such that x≻y then
      Ar:=Ar \ B∪{x}
    end if
  end for
  Ar' := {y∈Ar | ∄z∈Ar: z≻y}
  D := Ar \ Ar'
  if |Ar'| < μ then
    Fill Ar' with μ−|Ar'| individuals y∈D in increasing order
                                                of |{z∈Ar'∪D | z≻y}|
  end if
  Return: Ar'
```

The algorithm consists of two parts – insertion of individuals from *P* to *Ar* and selection of most promising part of *Ar* into *Ar'*. The insertion rules for ε-archive are straightforward:

- a new solution is added if the ϵ-box is empty
- a new solution replaces the old solution in the ϵ-box if the new solution strictly dominates the old one
- otherwise the new solution is omitted

The combination of the selection operator (Alg. 1) and the variation based on the probabilistic model results in a Bayesian Multi-objective Optimization Algorithm described in Alg. 2. In this ($\mu + \lambda$, ε) - BMOA, μ denotes the (minimum) number of parents that survive to the next generation being the input to build the model, λ the number of samples from the model in one generation and ε the factor that determines the granularity of the approximation.

```
Algorithm 2 (μ+λ,ϵ) - BMOA
while |Ar| < μ do
   Randomly create an individual x.
   Ar := Select(Ar, {x}, μ, ϵ)
end while
while Termination criteria not fulfilled do
   Create Bayesian Model M from Ar.
   Sample λ new individuals from M.
   Mutate these individuals and put them into B.
   Ar := Select(Ar,B,μ,ϵ)
end while
```

4.7 DEBOA

New object oriented rapid prototyping environment DEBOA fulfills the following goals:

- redesign of BOA core
- reusability of BOA core for variety of applications
- visualization of BOA evolution process
- interactive parameter settings

By combining the object-oriented design with the other Java features, DEBOA has several unique advantages:

- in the Java applet mode it is suitable for WWW presentations
- pre-compiled modules with new visualizations and new fitness computation can be dynamically loaded during run-time as new classes
- the class files are easily portable, no need to distribute the source code
- the class containing optimization core is reusable, it can be easily incorporated into the final project

For object oriented design the "design patterns" technique was used. The simplified scheme of DEBOA system is shown in the following UML diagram.

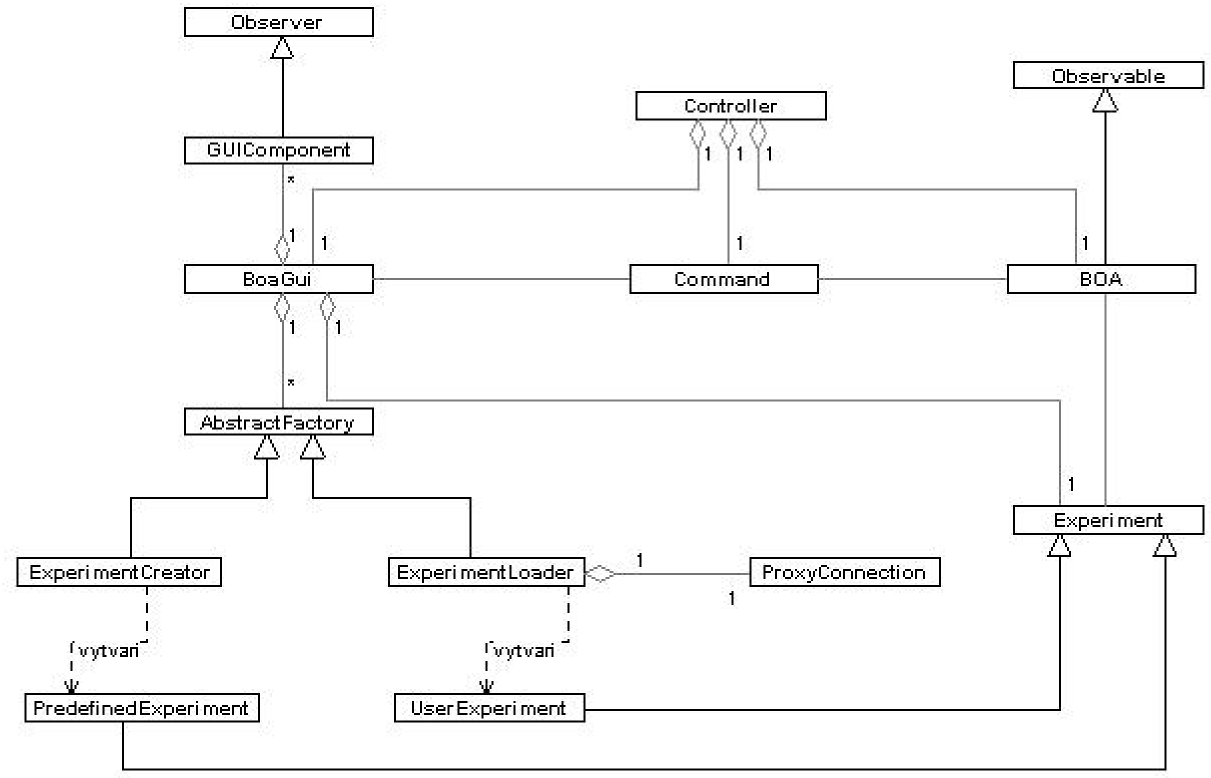


Fig. 4.1: The UML diagram of DEBOA classes

5 CONCLUSION AND FUTURE WORK

The main contributions and conclusions of this thesis can be summarized as follows:

- A new paradigm for Evolutionary Computation was explored and the new formal specification of EDA algorithm was formulated.

- It was developed a new probabilistic graphical model composed of binary decision trees with mixed decision nodes, which is more accurate and reliable model than the pure Bayesian network, more general model than Gaussian network and more useful for building blocks evolution than mixtures of Gaussians.

- A new advanced EDA algorithm MBOA (Mixed Bayesian Optimization Algorithm) for solving problems with real-valued parameters was designed and implemented. Its main advantage against the mostly used IDEA and EGNA approach is the backward compatibility with discrete domains, so it is uniquely capable of learning linkage between mixed continuous-discrete genes. MBOA handles the discretization of continuous parameters as an integral part of the learning process and the divide-and-conquer approach used for construction of decision trees exhibits high performance. For experiments with MBOA a new mixed continuous-discrete benchmark Fmixed was established, which is very hardly sensitive to correctness of decision tree building algorithm. MBOA was able to find global optimum, so the dependencies within each pair of deceptive mixed parameters have been successfully discovered.

- Original concept of parallelization of BN construction was proposed on the basis of predefined ordering of BN nodes. Parents for each node can be found separately without checking for acyclic dependency graph. This completely removes the need for communication between parallel processes and enables nearly linear speedup. It should be emphasized that the usage of concepts of parallel construction of probabilistic model is not limited only to the area of Estimation of Distribution Algorithm. The graphical probabilistic model became an important tool in the artificial intelligence, expert systems, data mining, etc.

- Three different versions of parallel EDA algorithms were proposed. The pipelined Parallel BOA algorithm (PBOA) was proposed for fine-grained type of parallelism with tightly connected communication channels. Simulation results showed that the quality of generated network was almost the same as in the sequential case. The Distributed Bayesian Optimization Algorithm (DBOA) was proposed and implemented for coarse-grained type of parallelism using the message passing communication (MPI). The experiments were successfully run on the uniform cluster of Sun Ultra 5 workstations with UltraSPARC II processors connected by fast Ethernet multi-port router. In the multithreaded MBOA algorithm an efficient parallel construction and sampling of binary decision trees was proposed and successfully validated via simulation in Transim tool.

- Additional methods for utilization of prior knowledge were suggested to speed up the convergence. In a Problem Knowledge-based BOA algorithm (KBOA) the partial (local) information about the problem was used to adjust the prior of probabilistic model and injection of building blocks was used to improve the quality of initial population.

- Multi-criterial BOA algorithms were designed. The first one was the Pareto BOA algorithm based on promising Pareto-strength fitness technique. It was comparable to the best recombination-based multi-objective algorithms of its time - NSGA and SPEA. More recent is the Bayesian Multi-objective Optimization Algorithm $(\mu+\lambda,\varepsilon)$-BMOA, which has been designed using a probabilistic model based on binary decision trees and a special selection scheme based on epsilon-archives. The convergence behavior of the algorithm can be tuned via the values of m, the minimal population size to estimate the probabilistic model, and ň, the approximation factor. The algorithm was empirically tested on different instances of the 0/1 multi-objective knapsack problem and successfully compared to NSGA-II and SPEA2 algorithms.

- Visual environment DEBOA for rapid prototyping of BOA optimization applications was developed. It includes all the important features of evolutionary toolkits – visualization, extensibility, reusability and portability. Due to using of Java language this system fulfills all the requirements for modern development system. One of its unique properties is the ability to be executed as java applet for WWW presentations. As far as I know, DEBOA is the only development environment based on Bayesian optimization algorithm for efficient solution of complex optimization problems with high epistasis.

The future work will be oriented towards the application of proposed parallel construction of dependence graph and usage of proposed mixed decision tree model and its metrics in non-evolutionary tasks like artificial intelligence, expert systems and data mining.

REFERENCES

[1] Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press, New York, Oxford, 1996.

[2] Bosman, P. A. N., Thierens., D.: Continuous iterated density estimation evolutionary algorithms within the IDEA framework, Proceedings of the Optimization by Building and Using Probabilistic Models OBUPM workshop, Genetic and Evolutionary Computation Conference GECCO-2000, pp. 197-200. 2000.

[3] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In Proceedings of the Parallel Problem Solving from Nature – PPSN VI, pp. 849-858, Springer-Verlag, 2000.

[4] Heckerman, D., Geiger, D., Chickering, M.: Learning Bayesian networks: The combination of knowledge and statistical data. Technical Report MSR-TR-94-09, Microsoft Research, Redmond, WA, 1994.

[5] Kostka R.: The development system for the class of Bayesian Optimization Algorithm, Faculty of Informaton Technology, VUT Brno, Brno, 2001.

[6] Laumanns, M., Thiele, L., Deb, K., Zitzler, E.: On the convergence and diversity-preservation properties of multi-objective evolutionary algorithms. Technical Report 108, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, 2001.

[6] Pelikan, M., Goldberg, D. E., Cantú-Paz, E.: BOA: The Bayesian optimization algorithm. Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99, volume I, pages 525-532, Orlando, FL, 1999. Morgan Kaufmann Publishers, San Fransisco, CA.

[8] Pelikan, M. A Simple Implementation of Bayesian Optimization Algorithm in C++(Version1.0). Illigal Report 99011, February 1999, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, Illinois, 1999.

[9] Pelikan, M., Goldberg, D. E., Tsutsui, S.: Combining the Strengths of the Bayesian Optimization Algorithm and Adaptive Evolution Strategies. IlliGAL Report No. 2001023, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, 2001.

[10] Srinivas, N., Deb, K.: Multiobjective Optimization using Nondominated Sorting in Genetic Algorithm. Evolutionary Computation, Vol.2, No.3, 1994, pp. 221-248.

[11] Tsutsui, S., Pelikan, M., Goldberg, D. E.: Evolutionary Algorithm Using Marginal Histogram Models in Continuous Domain. IlliGAL Report No. 2001019, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL, March 2001.

[12] Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications, PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[13] Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. EUROGEN 2001 - Evolutionary Methods for Design, Optimisation and Control, Athens, Greece, CIMNE, 2001.

ABSTRACT

The thesis deals with the new evolutionary paradigm based on the concept of Estimation of Distribution Algorithms (EDAs) that use probabilistic model of promising solutions found so far to obtain new candidate solutions of optimized problem.

There are six primary goals of this thesis:

1. Suggestion of a new formal description of EDA algorithm. This high level concept can be used to compare the generality of various probabilistic models by comparing the properties of underlying mappings. Also, some convergence issues are discussed and theoretical ways for further improvements are proposed.

2. Development of new probabilistic model and methods capable of dealing with continuous parameters. The resulting Mixed Bayesian Optimization Algorithm (MBOA) uses a set of decision trees to express the probability model. Its main advantage against the mostly used IDEA and EGNA approach is its backward compatibility with discrete domains, so it is uniquely capable of learning linkage between mixed continuous-discrete genes. MBOA handles the discretization of continuous genes as an integral part of the learning process, which outperforms the histogram-based approach. The original metrics for MBOA is derived as the incremental version of Bayesian Dirichlet metrics (BDe). Its usefulness for modelling of Boolean functions is also investigated and confirmed.

3. Parallelization of EDA algorithms. Different versions of parallel EDA algorithms are proposed for fine-grained, coarse-grained and multithreaded environment. All these algorithms use the original principle of restricted ordering of nodes in Bayesian network to minimize the communication between parallel processes:

- The pipelined Parallel Bayesian Optimization algorithm (PBOA) is proposed and simulated for fine-grained type of parallelism. The target platform for its implementation can be Transputers, or one-purpose MIMD processor.
- The Distributed Bayesian optimization algorithm (DBOA) is proposed and implemented for coarse-grained type of parallelism. The experiments are running on the uniform cluster of Sun Ultra 5 workstations connected by fast Ethernet switch. Both PBOA and DBOA are based on the parallelization of classical Pelikan's BOA code.
- The multithreaded MBOA algorithm, which uses original MBOA code with mixed decision trees model, was simulated in Transim tool.

In all parallel algorithms the additional progress towards realtime performance can be achieved by choosing the KBOA version of BOA algorithm. The concept of KBOA is based on the partial (local) information about the linkage. This information is used to adjust the prior of probabilistic model and injection of building blocks is used to improve the quality of initial population.

4. Extension of single-objective EDA to multi-objective EDA. The experiments show that the realized Pareto-strength BOA algorithm, which utilizes a promising Pareto-strength niching technique, outperforms the classical constraint- or weighting-based approach and is comparable to best recombination-based multiobjective-algorithms of its time - SPEA and NSGA. A completely new optimization technique for MBOA core was designed in collaboration with the author of SPEA2. This new selection operator based on epsilon-archives leads to implementation of Bayesian Multi-objective Optimization Algorithm (BMOA).

5. Design of the modular development system DEBOA for rapid prototyping of optimization applications on the basis of BOA algorithm. The general requirements on the development system are identified including modularity and visualization of results. The design of the development system itself is realized on the platform of Java language, which ensures great portability. DEBOA system can be easily extended to support new types of fitness functions as well as new types of visualizations.

6. Testing of the developed algorithms on well-known benchmarks as well as several real-world problems, mainly from the area of circuit design.

ABSTRAKT

Práce shrnuje pokročilé techniky genetické optimalizace založené na principu generování nových jedinců z pravděpodobnostního modelu rodičovské populace. Analýza současného stavu výzkumu v oblasti těchto EDA algoritmů (Estimation of Distribution Algorithm) vede k identifikaci několika aplikačních oblastí, v nichž principiálně nemohou být současné EDA optimalizátory uspokojivě aplikovatelné. Jedná se zejména o oblast řešení úloh se spojitými a smíšenými parametry, dále řešení rozsáhlých úloh a multikriteriálních úloh. Práce navrhuje nové metody a pravděpodobnostní modely překonávající uvedené bariéry.

Vzhledem k absenci formálního přístupu v literatuře je najprve navržen formální popis EDA algoritmu a jsou nastíněna teoretická východiska pro další vylepšení existujících EDA algoritmů.

Dále je práce věnována metodám řešení úloh se spojitými parametry. Je navržen nový pravděpodobnostní model založený na použití rozhodovacích stromů. Strukturálně se model podobá CART modelu používanému v oblasti dolování dat. K jeho konstrukci je však použitý Bayesovský přístup – z BDe metriky je odvozena nová inkrementální metrika pro ohodnocení kvality potenciálních rozhodovacích uzlů ve stromu. Výhodou nového modelu je zpětná kompatibilita s diskretními parametry. Navržený algoritmus MBOA (Mixed Bayesian Optimization Algorithm) je tedy vhodný i na řešení úloh se smíšenými parametry, na rozdíl od IDEA algoritmů (Iterated Density Estimation Algorithm) používajících pouze spojité pravděpodobnostní modely jako je Gausovská síť nebo Gausovská jádra. Další výhodou MBOA je, že diskretizace spojitých parametrů je integrální součástí konstrukce modelu, čímž se dosahuje lepších výsledků než v případě apriorní diskretizace pomocí histogramů.

Další kapitola je věnována podmínkám použití EDA na rozsáhlé problémy, tedy algoritmickému zjednodušení a paralelnímu zpracování. Jsou popsány nové metody hardwarové a softwarové akcelerace, zejména efektivní paralelní metoda konstrukce pravděpodobnostního modelu. Principem je dopředné stanovení topologického uspořádání uzlů v grafu závislostí. Každý procesor nemusí při konstrukci své části modelu komunikovat s ostatními procesory, neboť při přidávání hran je apriorně zajištěna acyklicita grafu. Tím klesá časová složitost konstrukce Bayesovské sítě lineárně s počtem použitých parallelních procesorů. Na uvedeném principu byly navrženy 3 paralelní EDA algoritmy.

Důsledný návrh všech částí EDA algoritmu podle zásad distribuovaného zpracování a jeho praktická implementace je výsledkem práce na optimalizátoru DBOA (Distributed Bayesian Optimization Algorithm). Ten byl odladěn v síti 10 pracovních stanic typu Sun Ultra 5 a používá komunikaci metodou zasílání zpráv pomocí standardní knihovny MPI (Message Passing Interface). Algoritmus konstrukce a vzorkování pravděpodobnostního modelu je založen na modelu distribuovaného zpracování typu "farmer-workers", takže je optimální výkonnosti dosaženo i v případě nehomogenní výpočetní sítě.

Také byla navržena a simulována varianta tohoto algoritmu pro jemnozrnnou úroveň granularity paralelismu, nazvaná PBOA (pipelined Parallel Bayesian Optimization Algorithm). Zatímco DBOA i PBOA používají jako pravděpodobnostní model Bayesovskou síť z klasické BOA a liší se hlavně ve způsobu generování nových jedinců, tak třetí paralelní EDA algoritmus je navržen speciálně pro pravděpodobnostní model založený na rozhodovacích stromech, tedy pro MBOA. Byl navržen vícevláknový paralelní algoritmus MBOA a jeho činnost byla úspěšně simulována pomocí nástroje TRANSIM.

Následující část práce je věnována optimalizaci multikriteriálních úloh. Byly navrženy dva multikriteriální EDA algoritmy. Prvním je Pareto-BOA používající principu Pareto síly pro ohodnocení jedinců v populaci. V rámci řešení grantu "Paralelizace Bayesovského Optimalizačního Algoritmu" byla naké navržena integrace Pareto-BOA s DBOA, která mimo jiné zahrnuje i algoritmus pro paralelní detekci a zpracování Pareto fronty. Druhým multikriteriálním EDA algoritmem je (μ+λ,ε)-BMOA (Bayesian Multiobjective Optimization Algorithm) navržený ve spolupráci s Marco Laumannsem z ETH Zurich. Na rozdíl od Pareto-BOA byl tento algoritmus cílený na pravděpodobnostní model s rozhodovacími stromy, tedy jedná se opět o rozšíření MBOA. Díky použití techniky epsilon-archivu je algoritmus teoreticky schopen garantovat nalezení globálního optima.

Na rozdíl od ostatních prací zabývajících se EDA nezůstaly testovací úlohy omezeny pouze na umělé aditivně-dekomponovatelné problémy, ale bylo ověřeno použití EDA algoritmů na množství praktických úloh, zejména z oblasti návrhu obvodů. Dalším směrem k praktickému nasazení EDA algoritmů je i vytvoření systému pro rychlé prototypování aplikací na bázi Bayesovských evolučních algoritmů. Program DEBOA je implementovaný v Jazyce JAVA a je spustitelný i jako internetová prezentace.