pMOHypEA: Parallel Evolutionary Multiobjective Optimization using Hypergraphs

Jörn Mehnen, Thomas Michelitsch,
Karlheinz Schmitt and Torsten Kohlen

No. CI-189/04

Technical Report ISSN 1433-3325 Dezember 2004

Secretary of the SFB 531 · University of Dortmund · Dept. of Computer Science/XI
44221 Dortmund · Germany

This work is a product of the Collaborative Research Center 531, "Computational Intelligence," at the University of Dortmund and was printed with financial support of the Deutsche Forschungsgemeinschaft.

pMOHypEA: Parallel Evolutionary Multiobjective Optimization using Hypergraphs

Jörn Mehnen[1], Thomas Michelitsch[1], Karlheinz Schmitt[2], and Torsten Kohlen

[1] University of Dortmund, Department of Machining Technology, ISF, Germany
mehnen@isf.de, michelitsch@isf.de, WWW home page: http://www.isf.de
[2] University of Dortmund, Chair of Systemanalysis, Germany
karlheinz.schmitt@cs.uni-dortmund.de, WWW home page:
http://ls11-www.cs.uni-dortmund.de

Abstract. An overview of the state-of-the-art in parallel evolutionary multiobjective optimization with a special view on the population structures is presented. An introduction to the theory of hypergraphs is given. The idea of hypergraphs is to scale freely between all existing population structures from coarse-grained island-models to fine-grained diffusion-models. Implementation details of pMOHypEA, a parallel evolutionary multiobjective optimization system that uses hypergraphs to structure populations, are given. First results of the concept are discussed.

1 Introduction

Population based optimization heuristics like evolutionary algorithms (EAs) or particle swarm optimization (PSO) in most cases require large amounts of fitness function evaluations. This can be problematic in practice, since the evaluations are often very time consuming, e.g. if finite element analyses (FEM) or complex simulations are used.

For parallelization purposes the use of a standard EA with equally distributed evaluations is adequate only if constant evaluation times can be assumed and if the processing nodes have the same performance. If the target architecture is a heterogeneous cluster of processing nodes or if the nodes are interconnected by a heterogeneous network, a more sophisticated distribution strategy is recommended to preserve a high efficiency. In order to achieve a good scalability and to reduce the communication overhead, it can be helpful to free the dependencies between the individuals and to utilize appropriately generalized EAs which support multiple temporarily independent populations. If treating complex multiobjective problems these generalized EAs often outperform standard EAs with a single homogenous population even if a single computer and no parallel architecture is used.

All parallel multiobjective evolutionary algorithms presently known utilize specific types of population neighborhood structures. Most commonly either regular grids or islandmodels are used.

A new and flexible hypergraph approach for the population neighborhood management will be introduced in this paper. In combination with population based optimization techniques it allows to adapt the population neighborhood structure to the problem's properties and to the underlying structure of a parallel computer system. A parallel multiobjective EA based on NSGA-II called pMOHypEA using a hypergraph approach has been implemented and tested.

The paper is organized as follows: The next section gives an overview of common parallelization approaches of evolutionary algorithms. A selection of parallel multiobjective EAs is discussed in Section 3. The hypergraph approach is subject of Section 4. In Sections 5 and 6 an implementation of a parallel hypergraph NSGA-II and the experimental setup are presented. The first test results are analyzed in Section 7. Finally a brief summary will be given and the conclusions will be highlighted.

2 Parallelization of Evolutionary Algorithms

2.1 Parallelization and Population Structure

Typically, the parallelization of evolutionary algorithms is realized by the design of specific mutation, recombination, evaluation and/or selection operators. The selection operator plays a key role, since multiple individuals are compared to each other with regard to their fitnesses. The sets of competing individuals arise from the structure of the population. In many cases the computation time and communication cost caused by the selection are dependent on the sizes and configurations of these sets. Recombination is the other operator which is directly influenced by the population structure. At least in the case of pairwise recombination the parallelization overhead is generally significantly lower than that of the selection.

The communication distance between two individuals is determined by their position in the population structure. It is possible to realize panmictic populations as well as populations which show a seldom, indirect and/or delayed genetic transfer. In the latter case a spacial or temporal separation can lead to an independent evolution within different subsets of the population for several generations. This usually results in a higher genetic diversity and can be advantageous especially if complex problems are to be solved.

Commonly the population structure does not match the communication structure of the parallel processor network on hardware level, but a certain correlation can be beneficial concerning the performance. A special kind of temporal separation can be achieved by an asynchronous parallel computation. The resulting asynchronous evolutionary optimization is comparable to the evolution in nature and its asynchronous character. Though algorithmically more complicated, the asynchronous parallel computation shows performance advantages especially in case of varying and unpredictable fitness evaluation costs or if heterogeneous parallel computer architectures are used.

2.2 Parallelization of MOEA

As an alternative to spacial separations which are derived from the communication structure, the individuals can explicitly be allocated to particular subpopulations depending on their affiliation to certain regions of the search space. In case of MOEA (multiobjective evolutionary algorithms) a disjoint decomposition of the objective space may be used. The main goals of parallelizing MOEA are usually the acceleration of the computation and the gain of quality according to the approximation of the true Pareto-Front or the diversity of the solutions. Research on the theoretic background of parallel MOEA (pMOEA) [19] is still quite rare. First theories regarding the population size [24] or regarding rather uncommon strategies [10] have been published.

2.3 Parallelization Approaches

Master-Slave-Approach. Here a single processor (master) takes control of the selection operator. Recombination, mutation and evaluation are executed by the other processors (slaves) independently. This approach is rather easy to implement especially if synchronized processes are used. It is used especially in the case where computation costs of the fitness function evaluations are high. Otherwise the communication can turn to be a bottleneck. The simple communication structure is advantageous if the processing power is distributed over the internet. The system GAIN (Genetic Algorithm running on the INternet) [11] was one of the first that made use of this multiobjective evolutionary approach. The approach was appliefd to several real world problems: e. g. compuational electromagnetics (CEM) and fluid dynamics (CFD) as well as aerodynamic issues [17] and X-ray Plasma Spectroscopy [9].

Island-Approach. Here the population is divided into spatially separated subpopulations (demes). The islands correspond to MOEA-instances that are running nearly independently. Genetic information is exchanged either by migration or pollination. In the first case an individual is transferred from the original deme to an other island in the neighborhood. In the second case the individual is left were it is and a copy is transferred. The amount of demes and their sizes as well as the interconnection structure and the migration rate are parameters of the island-model. The population structure and the hardware level architecture do not have to match. A certain amount of demes can be allocated to a single processor or vice versa. The following variants of parallel multicriterial island-models can be distinguished:

1. Identical MOEA with identical parameter settings are applied to every island (homogeneous island-model).
2. Identical MOEA with different parameter settings or completely different multicriterial EA are used. This approach is called heterogeneous. Both homogeneous and heterogeneous island-models are very common [19][21].
3. Different reduced or modified fitness function sets and corresponding singleobjective or multiobjective EA are assigned to every island. Examples for SPEA2 and NSGA-II are given in [14]. This approach is motivated by the idea of dividing a multiobjective problem into several reduced problems that can be solved step by step.

4. Every island represents a certain disjoint region of the search space or the objective space. This relatively new island-model approach is presented in [8]. The successful application to the test functions ZDT1 to ZDT3 and DTLZ2 is described in [2]. A regular assignment of the processors to certain ranges of the Pareto-front (niching) has been recognized.

Diffusion-model. Unlike the island-model, in the diffusion-model the individuals are not grouped in subpopulations. The communication relationships are defined on the individual level (fine-grained parallelism). Commonly von-Neumann or other local neighborhoods are defined for every individual in the same way. The communication structure of the whole population often shows a regular pattern (e. g. torus, grid). Recombination and selection operators access individuals in the direct neighborhood. Over the generations good solutions can diffuse throughout the grid. A temporal and spacial separation is caused by the high average communication distances. Therefore a diverse population is far more likely to emerge than island-models with panmictic subpopulations. Diffusion-models result in high communication costs. They can be reduced if subsets of adjoining individuals are allocated to single processors each. Multi-objective diffusion-models have been applied to combinatoric test functions [15] and specific real world problems. The static population structure can be replaced by a dynamic structure [20][12].

Hierarchic hybrid models. Here island type population structures are used in the top level. Certain MOEA are applied to every island. It is possible to recursively subdivide islands into smaller islands of new hierarchical levels. In this way a hierarchy structure can be realized. Cantú-Paz describes a singleobjective optimization algorithm [3]. A generalized multiobjective variant is presented in [19]. A hierarchical pMOEA with metaheuristics of the top levels tuning the parameters of the algorithms located on lower levels is introduced in [4].

3 Selected Parallel Multiobjective Evolutionary Algorithms

De Toro et al. introduce the PSFGA (Parallel Single Front Genetic Algorithm) that utilizes a master-slave-paradigm [17]. A sequential SFGA is executed on every processing unit. Exclusively non-dominating individuals are regarded for the selection and variation. A crowding-distance is taken into account in order to maintain the diversity of the approximation. The PSFGA has been tested on ZDT1-4 and 6 with the S-metric (hypervolume-metric) [22]. The authors prove a speed-up that increases with the number of processors up to a certain point. This saturation effect is typical for master-slave-models. It is caused by the rising communication overhead. If the amount of processors is very high, the speed up may even decline if further processors are attached. The authors show that the implemented SFGA exceeds the performance of the NSGA.

Xiong and Li [21] introduce the PSPMEA (Parallel Strength Pareto Multiobjective Evolutionary Algorithm) system, which is based on the SPEA2 [23] and utilizes the master-slave-approach as well as the island-model. The islands differ in the parameter settings for the recombination and mutation probabilities. Archives are used for the exchange of individuals between the islands. The quality of the Pareto-front approximation in comparison to the sequential SPEA2 is different but PSPMEA seems to do better on a certain complex combinatoric test problem.

Van Veldhuizen et al. present generic models for master-slave, island-model and diffusion-model MOEA [19]. They are joined in a generic system called pMOMGA-II (parallel Multi-Objective Messy Genetic Algorithm). A detailed comparison of the three parallelization paradigms still has to be done. The authors explain that the migration scheme has a great effect on the quality of the results. The comparison of the sequential and the parallel version of the MOMGA-II clarifies that complex problems are required for the parallel version to show its benefits.

No systematic analyses for diffusion-model based MOEA are known yet.

4 Parallel multiobjective Hypergraph EA

4.1 Motivation

The classification of parallel population based search heuristics according to the underlying population structure brings up the question of how the data exchange between the individuals should be structured. A systematic analysis of different population structures on multiobjective standard test problems could give insight to which parallelization paradigm should be applied in order to maximize the quality of the Pareto-front approximation or the convergence probability for a certain problem class. Most publications are restricted to analyses of certain algorithms and the comparison of parallel algorithms with others or their sequential counterparts. An inter-paradigm-analysis has not been found in literature yet. This paper will introduce a parallel MOEA that is capable of scaling between both extremes: the island-model and the diffusion-model. A special data structure – the hypergraph – for the definition of complex population structures will be utilized. The hypergraph is an abstract concept that is not restricted to a specific hardware structure.

4.2 Hypergraphs

All population structures, coarse-grained and fine-grained, can be modeled with the hypergraph [1] data model. Sprave has introduced this model for the theoretic analyses of a singleobjective EA applied to arbitrary population structures [16]. In the classic graph theory an edge of an undirected graph is defined as a connection between two vertices. Hypergraph edges (sometimes called hyperedges) can interconnect any sets of vertices with at least one element.

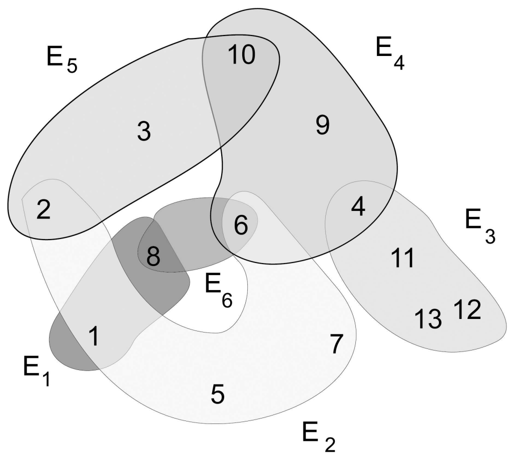

Fig. 1. Hypergraph with 6 edges E_1,...,E_6 and 13 vertices X_1,...,X_{13}.

Definition 1 *Hypergraph*
$X = \{x_1, x_2, \dots, x_n\}$ is a finite set of vertices. $E = (E_i | i \in I)$ is a family of subsets of X with the set of indices I. The family E is called hypergraph on X if

$$E_i \neq \emptyset \qquad \forall i \in I \tag{1}$$

$$\bigcup_{i \in I} E_i = X. \tag{2}$$

Figure 1 shows an example of a hypergraph with the vertex set $X = \{1, \dots, 13\}$ and the edges $E_1 = \{1, 8\}$, $E_2 = \{1, 2, 5, 6, 7\}$, $E_3 = \{4, 11, 12, 13\}$, $E_4 = \{4, 6, 9, 10\}$, $E_5 = \{2, 3, 10\}$ and $E_6 = \{6, 8\}$.

Classic graphs with two vertices per edge are often represented by adjacency lists or alternatively by incidence matrices. In the case of incidence matrices an edge i of the graph is represented by a corresponding row i. The element $a_{i,j}$ of the matrix is 1 if vertex j is element of the edge i ($X_j \in E_i$). Otherwise it is 0. With this definition incidence matrices can also be used to represent hypergraphs. The following 13×6-incidence matrix A corresponds to the hypergraph of figure 1.

$$A = \begin{pmatrix} 1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1\,1\,1 \\ 0\,0\,0\,1\,0\,1\,0\,0\,1\,1\,0\,0\,0 \\ 0\,1\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0 \end{pmatrix} \tag{3}$$

Hypergraphs can be used to model population structures:

Definition 2 *A population structure Π of a population P with the cardinality $|P| = \lambda$ is a pair (H, Q), consisting of a hypergraph $H = (X, E)$, $X = \{0, \ldots, \lambda - 1\}$* [3]*, $E \subseteq \mathcal{P}(X)$ and a partition $Q \subset \mathcal{P}(X)$ of X with $|Q| = |E|$. The hyperedge E_i is called subpopulation or deme of all elements $Q_i \in Q$.*

Based on this paradigm population structures used in MOEA can be defined as follows:

Panmictic Populations. Standard EA make use of unstructured populations. This is comparable to a trivial parallelization with a single island. The population structure Π_{pan} is defined by a triple of sets:

$$\begin{array}{ll} population\ structure: & \Pi_{pan} = (X, E, Q) \\ set\ of\ vertices: & X = \{0, \ldots, \lambda - 1\} \\ partition: & Q = (X) \\ set\ of\ edges: & E = (X) \end{array} \tag{4}$$

Island-model. Since individuals can move from one deme to another (migration) the hypergraph corresponding to an island-model must be dynamic. The following population structure Π_{migr} of a migration capable island-model is modeled by the population $P = (0, \ldots, \lambda - 1)$ of the size $\lambda = r\nu$ and containing the r subpopulations $Q_i = \{i\nu, \ldots, i\nu + \nu - 1\}$ which are of equal size ν as well as a set $M_{s \to t} \subset Q_s$ of migrants from Q_s to Q_t. If pollination instead of migration is used, the definition of the edges must be adopted appropriately.

$$\begin{array}{ll} population\ structure: & \Pi_{migr} = (X, E, Q) \\ set\ of\ vertices: & X = \{0, \ldots, \lambda - 1\} \\ partition: & Q = \{i\nu, \ldots, i\nu + \nu - 1\}, i = 0, \ldots, r - 1 \\ set\ of\ edges: & E = (E_0, \ldots, E_{r-1}) \\ edge\ (migration): & E_i = Q_i \cup \bigcup_{s=0}^{r-1} Ms \to i \setminus \bigcup_{t=0}^{r-1} Mu \to t, \\ & \quad Ms \to i \subset Q_s, Mu \to t \subset Q_i \\ edge\ (pollination): & E_i = Q_i \cup \bigcup_{s=0}^{r-1} Ms \to i \quad Ms \to i \subset Q_s \end{array} \tag{5}$$

Diffusion-model. It is also possible to represent fine-grained diffusion-models by hypergraphs in a compact manner. Structures like grids, tori and others are supported. A population $P = (0, \ldots, \lambda - 1)$ with a symmetric ring-shaped structure $P = (0, \ldots, \lambda - 1)$ and a neighborhood radius ρ results in:

$$\begin{array}{ll} population\ structure: & \Pi_{Ring} = (X, E, Q) \\ set\ of\ vertices: & X = \{0, \ldots, \lambda - 1\} \\ partition: & Q = (\{x_0\}, \ldots, \{x_{\lambda-1}\}) \\ set\ of\ edges: & E = (E_0, \ldots, E_{\lambda-1}) \\ edge: & E_i = (i - \rho, \ldots, i, \ldots, i + \rho), \quad i = 0, \ldots, \lambda - 1 \end{array} \tag{6}$$

[3] The indices are calculated *modulo* λ.

The Euclidean distance on the grid has been used here for the definition of the neighborhood radius ρ. More complex structures may require other metrics like the Manhattan-metric (based on the L_1-norm) or the amount of grid-points visited (based on the L_∞-norm).

Intermediate structures. The main advantage of the hypergraph concept is that arbitrary intermediate population structures can be modeled, too. It is e.g. possible to realize an island-model with additional diffusion characteristic or a model with several small panmictic islands arranged on a toric grid.

5 Hypergraph applied to a MOEA: pMOHypEA

A parallel multiobjective evolutionary algorithm with a hypergraph represented population structure (pMOHypEA) based on NSGA-II has been implemented. Since NSGA-II does not require archives the parallelization is quite intuitive. The amount of parallel NSGA-II processes required is equal to the amount of panmictic subpopulations defined by the hypergraph. In case of the diffusion-model this is the amount of individuals λ, in case of the standard island-model it is the amount of islands r.

Minor modifications to some of the NSGA-II operators are necessary, so that they are applied to the respective sets of individuals. Ranking and replacement of rank-dominating individuals as well as the calculation of the crowding-distance and the execution of the crowded tournament selection can intuitively be used for all sorts of coarse-grained and fine-grained population structures. As in the panmictic case the selection pressure depends on the ratio of parents and offsprings but interdependencies with the population structure are expected. Regarding the migration the hypergraph model is very flexible. Even complex migration schemes can be applied. It is e.g. possible to temporarily avoid migration between some islands and to utilize a clocked migration between others. On the algorithmic layer migrants can be requested from islands of the neighborhood. These will respond by transmitting genetic data.

The parallelization concept of the implemented pMOHypEA is independent from the hardware. The population structure is mapped to the processors on the basis of user-defined instructions. For performance reasons the specific hardware architecture should be taken into account.

For comparison with most of the results given in the literature the S-metric is used. Alternatively the R-metric or the MMBBH-metric [13] can be used in order to achieve a better statistical relevance. Van Veldhuizen et al. point out that today's metrics are designed for the comparison of sequential algorithms. Special metrices for parallel algorithms are not known.

6 Experimental Setup

In the experiments the parameter settings of table 1 have been used. The basic algorithm parameters are identical with the parameters of a conventional NSGA-

II. The number of fitness function evaluations have been set constant for all experiments in order to guarantee a fair comparison of the results. The standard test functions ZDT1 and ZDT4 have been chosen. Both functions are commonly used in literature in sequential and parallel variation [8][21] and the functions are a sufficiently complex to be a challenge for parallel approaches. In order to compare the results with the literature, the well known S-metric of Zitzler [22] has been used.

Table 1. Specifications of the NSGA-II-parameters of pMOHypEA.

Symbol	ZDT1	ZDT4	Description
p_c	0.9	0.9	crossover probability
p_m	0.03	0.1	mutation probability
d_c	10	10	distribution index for the crossover
d_m	50	50	distribution index for the mutation
g	250	250	number of generations
n	30	10	dimension of the problem
s_t	tournament selection		type of selection
c_t	simulated Binary Crossover		type of crossover

The population structures have been designed according to the definitions shown in table 2. A simple panmictic island-model is described by the data in the first line. In a panmictic model all individuals can exchange their genetic material with each other. Therefore, in the respective model all individuals are completely interconnected with each other. The panmictic model is structurally identical to the common NSGA-II.

In the second, third, and forth line distributed island-models are described. The islands have been arranged on a ring. Between each two interconnected islands migrants are exchanged according to the migration rate and migration frequency. Pollination and migration have been investigated separately.

A diffusion-model is described by the last two lines. The upper line models a 14×14 torus shaped grid net with a von-Neumann topology, i. e. each individual has a neighbor to the left, to the right, above and below. The genetic material for recombination is exchanged only within this small region. Of course the regions around the individuals overlap. Therefore, a communication between individuals all throughout the net is possible. The last line models a special diffusion-model, i.e. a population structure with single individuals arranged on a ring with two neighbors each. This model has the longest possible communication distance between two individuals. In the diffusion-models no "real" migration rate nor frequency can be defined, although the data exchange is technically realized that way. Therefore, the values in table 2 are undefined.

The experiments were performed on a parallel and a sequential machine. The parallel computer was a SGI Origin 2000 (16 MIPS R10000 processors 196 MHz, 4 GB local shared memory and 789 MB peak band width). The one processor machine was a standard PC under LINUX.

Table 2. Specification of the population structure.

model	subpopulations	indiv. per subpop.	migration rate	migration frequency
panmictic	1 (1 CPU)	200	-	-
island ring	2 (2 CPU)	100	30	5
island ring	3 (3 CPU)	100	30	5
island ring	4 (4 CPU)	50	15	5
island ring	10 (10 CPU)	20	6	5
diffusion-model	196 (14 CPU)	1	-	-
diffusion-model	196 (14 CPU)	1	-	-

7 First Results

All experiments are performed on the parallel computer and on the sequential one processor PC. The parallel version of the pMOHypEA is asynchronous, i. e. the evaluation of each individual or island is done without waiting for the execution of the other elements. The asynchronous communication between the populations is supported by MPI. A quasi synchronous execution of the program is introduced by execution of the software on the one processor machine. An explicit synchronization was not needed. All experiments were repeated five times in synchronous and five times in asynchronous mode. The average values of the S-metric of the five runs are analyzed. Standard deviations and execution times are added where necessary.

7.1 Speed Up

The experiments with the asynchronous and synchronous versions all run stable. This cannot be expected naturally in asynchronous situations. Communication bottle-necks and slightly different execution times of single processes can yield instabilities of the system behavior. Slight fluctuations in the processor loads due to exchange problems of individuals were compensated by faster or slower evaluation of the available individuals in the populations. The results was a globally stable process.

One can see from table 3 that the parallelization generally yielded a relevant speed up. According to Ahmdals law the absolute speed up can be calculated via the equation

$$S_I(N) = \frac{T(1)}{T(N)} = \frac{T_s + T_p}{T_s + T_p/N} \tag{7}$$

where N is the number of islands/processes. $T(1)$ denotes the total time needed by one processor in sequential mode. This time can be separated in a serial part T_s and a parallelizable part T_p. Figure 2 shows the absolute speed up for the synchronous and the asynchronous mode. Due to the fact that the numbers of the individuals decrease with the number of islands, a fair speed up value $S_P(N) = S_I(N)/N$ was calculated.

Table 3. Average run times in seconds $\overline{s}$ and the corresponding standard deviations σ_s of pMOHypEA applied to ZDT1 and ZDT4 in asynchronous and synchronous mode.

	asynchronous ZDT1		synchronous ZDT1		asynchronous ZDT4		synchronous ZDT4	
Nr.	$\overline{s}$	σ_s	$\overline{s}$	σ_s	$\overline{s}$	σ_s	$\overline{s}$	σ_s
1	198.0	0.000	82.4	2.1	189.6	0.894	80.2	12.317
2	58.6	0.548	45.4	0.5	54.8	0.447	30.8	16.185
3	28.0	0.000	38.8	1.8	24.6	0.548	30.2	2.775
4	17.8	0.447	35.8	2.8	15.4	0.894	22.0	0.707
5	9.0	0.707	21.6	1.1	3.4	0.548	11.2	0.447

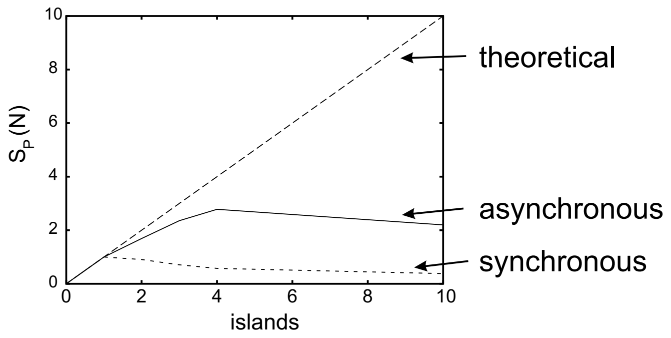

Fig. 2. The fair $S_P(N)$ speed up values for the asynchronous and the asynchronous approach applied to ZDT1.

Figure 2 shows a typical speed up graph. In the beginning in the parallel asynchronous version the speed up rises strongly and declines due to an increasing communication overhead. A maximum speed up is expected in the region of about 4 to 6 islands. The synchronous version reaches its fair speed up maximum in the panmictic case. This is behavior can be expected, because any additional communication reduces the speed of the one processor system.

7.2 Analyses of the Population Structures

In the experiments of [8] a fixed predefined hypervolumes value has to be reached. The number of generations that are sufficient to reach a value of 0.794 were counted. The reference point is set to $(1.0646, 1.0646)^T$ for both problems ZDT1 and ZDT4. The theoretically possible value for ZDT4 is 0.8. A maximum number of 250 generations are allowed.

Results for ZDT1 In table 4 the results for the analyses of the pMOHypEA applied to ZDT1 are shown. In case of the island-models the Pareto-Front was found by the pMOHypEA. The graphs of the Pareto-fronts are close to each other. The quality of the approximations did not significantly depend on the population structure. Only the linear ring and the diffusion-model show an insufficient spread of the Pareto-front. This may be due to the usage of the crowding distance that is not optimal for very small populations. Generally the Pareto-Front is covered quite good by the other models. A parallel approach to solve the problem ZDT1 seems neither necessary nor very sensible.

Table 4. Results of pMOHypEA applied to problem ZDT1.

Exp.	*parents*	*islands*	*migr.rate*	*migr.freq.*	$\bar{S}_{async}$	$\bar{S}_{sync}$	$\bar{gen}_{async}$	$\bar{gen}_{sync}$
1	200	1	0	0	0.797	0.797	44.8	45.6
2	200	2	30	5	0.797	0.797	55.6	55.6
3	198	3	20	5	0.796	0.796	57.8	60.4
4	200	4	15	5	0.796	0.796	69.8	65.8
5	200	10	6	5	0.795	0.795	97.4	93.8
6	196	196	-	-	-	0.133	250	250.0
7	196	196	-	-	-	0.452	250	250.0

Results for ZDT4 More interesting results have been gathered by the application of pMOHypEA to the more complex ZDT4 problem (see table 5).

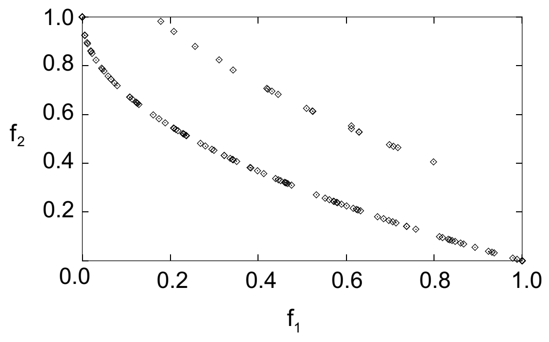

Fig. 3. Pareto-front found by pMOHypeEA with 15 islands applied to ZDT4.

All models show only a small variation of the average calculated S-metric value. Only the asynchronous solution shows a slightly higher variation $\sigma(S_{async})$ than the synchronous model. This indicates a certain influence of the asynchronous evaluation on the results because these variations cannot be seen in the synchronous case.

Differences in the results between the asynchronous and the synchronous solution are not very strong. Outliers in the results are due to premature stagnation of the algorithms or caused by not fully spread Pareto-fronts. More important

Table 5. S-metric values and the number of generations needed by the pMOHypEA to reach a predefined value when applied to ZDT4.

Nr.	$\sigma(S_{async})$	$\bar{S}_{async}$	$\bar{S}_{sync}$	$\sigma(S_{sync})$	$\sigma(gen_{async})$	$\bar{gen}_{async}$	$\bar{gen}_{sync}$	$\sigma(gen_{sync})$
1	0.080	0.761	0.670	0.133	28.6	212.6	219.8	41.4
2	0.080	0.760	0.689	0.099	97.2	171.2	205.8	78.9
3	0.233	0.796	0.796	0.076	90.5	187.6	156.8	88.9
4	0.325	0.650	0.724	0.099	81.7	162.6	190.6	82.5
5	0.081	0.759	0.683	0.167	70.2	149.8	171.6	75.9
6	-	-	0.179	0.187	0.0	250.0	250.0	0.0
7	-	-	0.101	0.115	0.0	250.0	250.0	0.0

is the fact that the asynchronous algorithms found the solution more often and also earlier than the synchronous counterpart.

In the first experiments a completely interconnected island-model was tested. The motivation for this is to compare pMOHypEA with the standard NSGA-II in the synchronous and asynchronous modes, i. e. the program was executed on the parallel computer and on the PC. A relevant influence of the hardware in the panmictic case cannot be seen.

In the model with two islands in the asynchronous case the migration model performs a little bit better than the pollination model. Generally a slight improvement of the convergence properties with respect to a conventional NSGA-II can be stated. The solutions of each of the two islands are evenly distributed over the Pareto-Front. A separation of the island solutions into local clusters cannot be noticed.

In the model with three islands a higher convergency probability was measured than that of a conventional NSGA-II. Pollination performed generally better than migration. The pollination seems to compensate the fast evolution of dominant suboptimal solutions and, therefore, avoids stagnation in local pre-Pareto-fronts.

The model with four islands behaves similar to the model with three islands. The Pareto-Front is covered very uniformly. This is true for the solutions of all islands.

The model with ten islands emphasizes the trend that pollination yields better results. Low migration rates should be preferred. Generally, the first analyzes suggest that the number of migrants should be chosen approximately proportional to the size of the islands. A higher number of islands also leads to a more steady convergence behavior of the hypervolume values. Migration leads to higher convergence fluctuations when small population sizes are used. A migration frequency of 5 and a pollination frequency of 10 seem to be a good choice for small islands. Figure 3 shows the Pareto-front found by the asynchronous 10 islands model.

First results using a model with 15 islands shows an even more improved convergence behavior. The Pareto-Front approximation after 250 generations is often better than runs with the conventional NSGA-II. Using pollination the front is reached in 90 percent of the cases, while NSGA-II has a change of

50 percent. The domination of single individuals can be avoided with a higher chance with an increased partitioning of the population structure.

The diffusion-models with 200 individuals/islands show a maximum of separation of the individuals. The general trend to better solutions and higher convergence probabilities with increasing numbers of islands is not continued. These models often yielded results with only one ore two extremal solutions. These results motivate the idea that the genetic operators of the NSGA-II are not appropriate for very small population sizes. First experiments with 100 islands show similarly insufficient solutions. Models with 70 islands definitely yielded better results.

8 Summary and Conclusion

A new parallel multiobjective evolutionary algorithm pMOHypEA is introduced. The system is using a hypergraph that allows to scale the population structures freely between coarse-grained island-models to fine-grained diffusion-models. The experiments are performed on a parallel computer and a PC in asynchronous and synchronous mode, respectively. The results show, that not only a significant speed up is gained. Depending on the problem, the convergence probability and approximation quality and spread of the Pareto-front solutions can be improved by using parallel models with significantly many islands. This improvement has a maximum and decreases for the diffusion-model. For small population sizes the genetic operators, that were adopted from a NSGA-II, have to be adapted adequately.

References

1. Berge, C.: Graphs and Hypergraphs. North-Holland mathematical library, North-Holland, Amsterdam, NL (1973)
2. Branke, J., Schmeck, H., Deb, K., Reddy, M.: Parallelizing Multi-Objective Evolutionary Algorithms: Cone Separation. Congress on Evolutionary Computation (CEC'2004), IEEE Service Center, Piscataway, NJ (2004) 1952–1957
3. Cantú-Paz, E.: Efficient and Accurate Parallel Genetic Algorithms. Genetic Algorithms and Evolutionary Computation, Kluwer Academic Publ. (2000)
4. Caswell, D. J.: Active Processor Scheduling Using Evolutionary Algorithms. Air Force Inst. of Tech. Wright-Patterson, Technical Report afit/gcs/eng/02-36, Dayton, OH (2002)
5. Coello Coello, C. A., van Veldhuizen, D. A., Lamont, G. B.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, New York, NJ (2002)
6. Collett, Y., Siarry, P.: Multiobjective Optimization. Principles and Case Studies. Cost Engineering in Practice, Speringer, Berlin (2003)
7. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Wiley-Interscience Series in Systems and Optimization, John Wiley, Chichester, UK (2001)

8. Deb, K., Zope, P., Jain, A.: Distributed Computing of Pareto-Optimal Solutions Using Multi-Objective Evolutionary Algorithms. Kanpur Genetic Algorithms Laboratory (KanGAL), Technical Report, 2002008 (2002)
9. E. Golovkin, I., J. Louis, S., C. Mancini, R.: Parallel Implementation of Niched Pareto Genetic Algorithm Code for X-ray Plasma Spectroscopy, Late Breaking Papers at the 2000 Genetic and Evolutionary Computation Conference, IEEE Service Center, Las Vegas, NV (2000) 222-227
10. Jansen, Th., P. Wiegand, R.: Sequential versus parallel cooperative coevolutionary (1+1) EAs, Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003), Canberra, Australia, IEEE Press, **4** (2003) 30–37
11. J. Stanley, T., Mudge, T.: A Parallel genetic Algorithm for Multiobjective Microprocessor Design. Proceedings of the Sixth International Conference on Genetic Algorithms, L.J. Eshelman et al. (eds.), Morgan Kaufmann Publishers, San Mateo, CA (1995) 597–607
12. Li, X., Kirley, M.: The Effects of Varying Population Density in a Fine-grained Parallel Genetic Algorithm. Proceedings of Congress on Evolutionary Computation, Hawaii (CEC2002), IEEE Service Center, Piscataway, NJ, **2** (2002) 1709–1714
13. Mehnen, J., Michelitsch, Th., Bartz-Beielstein, Th., Henkenjohann, N.: Systematic Analyses of Multi-objective Evolutionary Algorithms Applied to Real-World Problems Using Statistical Design of Experiments., 4th CIRP International Seminar on Intelligent Computation in Manufacturing Engineering, Italy, (2004) 171–178
14. Okuda, T., Hiroyasu, T., Miki, M., Watanabe, S.: DCMOGA: Distributed Cooperation model of Multi-Objective Genetic Algorithm. MPSN-II, The Second Workshop on Multiobjective Problem Solving from Nature, Granada, Spain (2002)
15. Rowe, J., Vinsen, K., Marvin, N.: Parallel GAs for Multiobjective Functions. Proceedings of the Second Nordic Workshop on Genetic Algorithms and Their Applications (2NWGA), J.T. Alander (ed.), University of Vaasa, Finland, (1996) 61–70
16. Sprave, J.: Ein einheitliches Modell für Populationsstrukturen in Evolutionären Algorithmen. PhD thesis, Universität Dortmund, Germany, (1999)
17. de Toro, F., Ortega, J., Ros, E., Mota, S., Paechter, B., Martin, J.: PSFGA: Parallel processing and evolutionary computation for multiobjective optimisation. Parallel Computing,**30** (2002) 5-6 721–739
18. van Veldhuizen, D. A.: Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. PhD thesis, Air Force Institute of Technology, OH (1999)
19. van Veldhuizen, D. A., Zydallis, J. B., Lamont, G. B.: Considerations in engineering parallel multiobjective evolutionary algorithms. IEEE Trans. on Evol. Comp., **7** (2003) 2 144–173
20. Weinert, K., Mehnen, J., Rudolph, G.: Dynamic Neighborhood Structures in Parallel Evolution Strategies., Complex Systems **13** (2002) 3 227–244
21. Xiong, S., Li, F.: Parallel Strength Pareto Multi-objective Evolutionary Algorithm for Optimization Problems. Proceedings of the 2003 Congr. on Evolutionary Computation, Canberra, AU, IEEE Press, Piscataway, NJ **4** (2003) 2712–2718
22. Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications. Shaker Verlag, Aachen, PhD thesis, ETH Zrich (1999)
23. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Communication Networks Lab (TIK), ETH Zürich (2001)
24. Zydallis, J. B.: Explicit building block multiobjective genetic algorithms: Theory, analyses, and new innovations Detp. of Elect. Comput. Eng., Air Force Inst. of Technology, PhD thesis (2003)