Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors

Joshua D. Knowles and David W. Corne

August 20, 2002

Abstract

Search algorithms for Pareto optimization are designed to obtain multiple solutions, each offering a different tradeoff of the problem objectives. To make these different solutions available at the end of an algorithm run, procedures are needed for storing them, one by one, as they are found. In the simple case this may be achieved by placing each point (the image of a solution in objective space) that is found into an 'archive' which maintains only nondominated points and discards all others. However, even a set of mutually nondominated points is potentially very large (infinite in continuous objective spaces), necessitating a bound on the archive's capacity. But with such a bound in place it is no longer obvious which points should be maintained and which discarded; we would like the archive to maintain a representative and well-distributed subset of the points generated by the search algorithm, and also that this set converges. To achieve these objectives we propose an adaptive archiving algorithm, suitable for use with any Pareto optimization algorithm, which has various useful properties as follows. It maintains an archive of bounded size, encourages an even distribution of points across the Pareto front, it is computationally efficient, and we are able (with caveats) to prove convergence. Previously proposed archiving algorithms, which we also discuss, have more general convergence properties, but at the expense of not being able to maintain an even distribution of points along the front, or are very computationally expensive, or do not guarantee to maintain a certain *minimum* number of points in the archive. In contrast, the method proposed here maintains evenness, efficiency, and cardinality, and provably converges under certain conditions (e.g. when there are two objectives) but not all. Finally, the notions underlying our convergence proofs support a new way to rigorously define what is meant by "good spread of points" across a Pareto front, in the context of grid-based archiving schemes. This leads to proofs and conjectures applicable to archive sizing and grid-sizing in any Pareto optimization algorithm maintaining a grid-based archive.

1 Introduction

In a review of multiobjective evolutionary algorithms (MOEAs) [Horn, 1997], Horn noted (page 6) that all practical implementations of MOEAs for Pareto optimization can be assumed to maintain (off-line) an archive of the best (nondominated) solutions found during an algorithm run. The archive is needed because the on-line population of a MOEA is usually of a very limited size and Pareto optimal solutions in it can be lost through stochastic selection processes. The observation of Horn is reflected in much MOEA literature where results are frequently presented in terms of the off-line solutions found. However, despite the widespread use of off-line storage of solutions, there has been little consideration of *how* best to implement it. In particular, how should we bound the size of the archive, whilst maintaining a representative sample of all the (nondominated) solutions found?

Although the problem of maintaining an archive may not have been too serious in the past, as MOEAs are applied to larger problems (with perhaps more objectives) it will become more important to bound archive size. Moreover, the advent of elitist MOEAs has meant that the archive is often, in reality, used as a secondary population, providing a source for the generation of new solutions. In this context, it becomes essential to bound the size of the store of solutions, whilst maintaining its representative nature, so that selection mechanisms can run effectively and efficiently.

With these considerations in mind, it is clear that some means of reducing the number of stored nondominated solutions is needed if we are to maintain and/or utilize an archive efficiently. However, we would also like our archive to have guaranteed convergence to a set which, in a well-defined sense, approximates the whole, true Pareto front. Thus, the use of simple clustering methods to reduce the set may not be sufficient for these purposes. To ensure convergence, more advanced schemes are needed. Convergence proofs for a number of MOEAs were given in [Hanne, 1999, Rudolph and Agapie, 2000] but the schemes under consideration in these papers do not aim to generate a bounded set which is a good approximation to the whole Pareto front. However, a more recent technical report by Laumanns *et al.* [Laumanns et al., 2001] proposed several schemes for maintaining a bounded archive which converges to a well-defined approximation of the whole Pareto front, although these methods have certain problems concerning the initial setting of important parameters. We will review this report in Section 6, and relate it to the work presented here.

The remainder of this paper is organized as follows. Section 2 gives some preliminary information and definitions needed for the analysis of the archiving algorithms. Section 3 describes three algorithms which all converge to a subset of the true Pareto front under certain conditions. However, each of these algorithms has drawbacks, which are discussed in this section. An adaptive grid archiving algorithm (AGA) is proposed in Section 4, and proofs relating to its convergence are given. Section 5 indicates the conditions when the

grid boundaries of the adaptive grid archiving algorithm does not converge, and demonstrates why these conditions prevent convergence. In Section 6, a brief description of the archiving algorithms recently proposed by Laumanns *et al.* is given. These algorithms are compared with the AGA and we draw some parallels between them, particularly the problems that all the algorithms have in providing an effective adaptive scheme which also guarantees convergence. Section 7 summarises and concludes.

2 Preliminaries

In the following we analyse the convergence properties of several archiving algorithms. We take, as the basis of our analysis, a model in which at every iteration of the archiving algorithm, a separate *generating process* generates a multiobjective vector (point) which must be archived. All proofs of convergence presented here rely on the fact that at *every* time step the generating process gives every point in the search space a non-zero probability of being generated. In practice, this assumption will be true whenever, for example, a mutation is applied to every bit in a binary string with some small probability, the standard method of generating a new point in a random mutation hillclimber.

Definition 2.1 (Objective Space) *The finite set Z denotes the* objective space *of all feasible objective vectors $\mathbf{z} \in \Re^K$, $K \geq 2$.*

We assume, without loss of generality, a minimization problem. Thus the dominance relation is defined as follows.

Definition 2.2 (Dominance Relation) *Let $\mathbf{z}^1 = \{z_1^1, z_2^1, \ldots, z_K^1\}$, $\mathbf{z}^2 = \{z_1^2, z_2^2, \ldots, z_K^2\}$ and $\mathbf{z}^1, \mathbf{z}^2 \in Z$. Then $\mathbf{z}^1$ dominates $\mathbf{z}^2$, denoted as $\mathbf{z}^1 < \mathbf{z}^2$, iff $\forall k \in 1..K, z_k^1 \leq z_k^2 \wedge \exists k \in 1..K, z_k^1 < z_k^2$.*

Definition 2.3 (Pareto Front) *The finite set $Z^* = \{\mathbf{z}^* \in Z \mid \nexists \mathbf{z} \in Z,\ \mathbf{z} < \mathbf{z}^*\}$ denotes the Pareto front of Z.*

Definition 2.4 (Archive) *The finite set $M_t \subseteq Z$ denotes the archive of objective vectors at time t.*

Definition 2.5 (Generating Process) *The process $\mathsf{Gen}(t)$ generates solutions from Z to be stored in M_t. The output of $\mathsf{Gen}(t)$ at time t is $\mathbf{z}_t$, and associated with $\mathsf{Gen}(t)$ there is a probability $\mathrm{pr}(t, \mathbf{z})$ of generating solution $\mathbf{z} \in Z$ at time t. $\mathsf{Gen}(t)$ has the property that $\forall t, \forall \mathbf{z} \in Z, \mathrm{pr}(t, \mathbf{z}) > 0$.*

Definition 2.6 (Nondominated set filter) *Given any set of objective vectors, Z^1:*
$\mathsf{ND}(Z^1) = \{\mathbf{z}^i \in Z^1 \mid \nexists \mathbf{z}^j \in Z^1,\ \mathbf{z}^j < \mathbf{z}^i,\ i, j \in 1..|Z^1|\}$.

Definition 2.7 *We also define the following relations between a vector* $\mathbf{z}^a$ *and a nondominated front* $Z_{nd} = \mathsf{ND}(Z_{nd})$*:*

$$\mathbf{z}^a < Z_{nd} \Longleftrightarrow \exists \mathbf{z} \in Z_{nd},\ \mathbf{z}^a < \mathbf{z}, \tag{1}$$

$$\begin{aligned} Z_{nd} < \mathbf{z}^a &\Longleftrightarrow \exists \mathbf{z} \in Z_{nd},\ \mathbf{z} < \mathbf{z}^a,\ \textit{or in other words} \\ Z_{nd} < \mathbf{z}^a &\Longleftrightarrow \mathsf{ND}(\{\mathbf{z}^a\} \cup Z_{nd}) = Z_{nd} \ \wedge\ \mathbf{z}^a \notin Z_{nd}, \end{aligned} \tag{2}$$

$$\mathbf{z}^a \sim Z_{nd} \Longleftrightarrow \mathbf{z}^a \not< Z_{nd} \ \wedge\ Z_{nd} \not< \mathbf{z}^a \ \wedge\ \mathbf{z}^a \notin Z_{nd}. \tag{3}$$

Finally we define the weak outperformance relation([Hansen and Jaszkiewicz, 1998]) between two nondominated sets:

Definition 2.8 (Weak outperformance) $A\ O_W\ B \Longleftrightarrow \mathsf{ND}(A \cup B) = A$ *and* $A \neq B$*. In other words, approximation* A weakly outperforms *approximation* B *if all points in* B *are 'covered' by those in* A *(where 'covered' means is equal to or dominates) and there is at least one point in* A *that is not contained in* B*.*

We analyse the different archiving algorithms that we shall consider by using a generic archiving algorithm $AA_{\mathsf{Reduce} \in RED}$ shown in Figure 1. At each iteration this algorithm generates one point and updates the nondominated solutions archive as specified by the Reduce function. A particular archiving algorithm may be specified from $AA_{\mathsf{Reduce} \in RED}$ by specifying a particular Reduce function from the set RED of all Reduce functions.

Definition 2.9 (Convergence) *We say that* M_t *is converged under algorithm* $AA_{\mathsf{Reduce} \in RED}$ *if it is in a state in which its members will not change for all future iterations, i.e., if* $\forall t > t_n,\ M_t = M_{t_n}$ *then* M_{t_n} *is converged. We also use the phrase: "*$AA_{\mathsf{Reduce} \in RED}$ *converges" under rule* Reduce*, as shorthand for saying that the archive of* $AA_{\mathsf{Reduce} \in RED}$ *converges.*

3 Three Convergent Archiving Algorithms

3.1 Unbounded archiving

Consider the simple Reduce function, Unbounded of Figure 2. At each step of the the resulting archiving algorithm, $AA_{\mathsf{Unbounded}}$, a new vector is accepted into the archive if it is nondominated amongst the current archive. If it dominates any members of the archive, the dominated members are removed. Clearly this algorithm maintains an archive of nondominated vectors. It is also trivial to show that this algorithm converges to the true Pareto front, given our assumptions about the generating function, and the finite

nature of Z.

The problem with the Reduce function of Figure 2 is that the size of the archive is bounded only by Z. In general Z may be very large and we would like to obtain only a sample of points from it.

3.2 Simple bounded archiving

To obtain a simple bounded archive we could modify the Reduce function given above to obtain the Reduce function, Bounded, shown in Figure 3. Here, a bound on the archive, *arcsize*, is chosen before running the algorithm. Once the archive reaches this bounding size, only dominating vectors are accepted, thus preventing exceeding the bound. The resulting algorithm AA_{Bounded} is similar to the archiving algorithm proposed by Rudolph and Agapie in [Rudolph and Agapie, 2000]. Given the assumptions of a finite Pareto set and a positive definite generating function, this algorithm converges to a subset of the Pareto front[1]. However, this algorithm does not encourage the storage of a good distribution of vectors. In fact, $AA_{\mathsf{Unbounded}}$ is efficiency preserving [Hanne, 1999] once the archive reaches its capacity bound. That is, new points can only enter the archive if they dominate one or more of the previously stored points. A consequence of this is that given a collection of vectors in the archive at time t, some regions of the Pareto front may not be reachable, even if none of the vectors present in the archive at time t is Pareto optimal. This is illustrated in Figure 4. In order to avoid this limitation one needs to relax the rule that nondominated vectors can never be removed from the archive.

3.3 $\mathcal{S}$ metric archiving

In the final archiving algorithm considered in this section, we include a rule to allow nondominated vectors to be removed from the archive, to make way for new ones which may improve the distribution of points. The new rule makes use of the $\mathcal{S}$ metric: a measure of the hypervolume in objective space that is dominated by a set of nondominated points. The $\mathcal{S}$ metric was originally proposed by Zitzler and Thiele [Zitzler, 1999]. We give here the formulation of it necessary for use with minimization problems.

[1] The proof is simple, but we omit it here for brevity.

The $\mathcal{S}$ metric

Given a point $\mathbf{z}^1 = \{z_1^1, z_2^1, \ldots, z_K^1\}$ in objective space Z, and a reference vector $\mathbf{z}^{\mathbf{ref}} = \{z_1^{ref}, z_2^{ref}, \ldots, z_K^{ref}\}$ dominated by $\mathbf{z}^1$, let the region dominated by $\mathbf{z}^1$ and bounded by $\mathbf{z}^{\mathbf{ref}}$ be defined as the set:

$$R(\mathbf{z}^1, \mathbf{z}^{\mathbf{ref}}) \triangleq \{\mathbf{y} \mid \mathbf{y} < \mathbf{z}^{\mathbf{ref}} \text{ and } \mathbf{z}^1 < \mathbf{y},\ \mathbf{y} \in \Re^K\}. \tag{4}$$

For a nondominated set A of vectors $\mathbf{z}^i, i = 1..|A|$, and a reference vector $\mathbf{z}^{\mathbf{ref}}$, that is dominated by all members of A, the region dominated by A and bounded by $\mathbf{z}^{\mathbf{ref}}$ is defined as the set:

$$R(A, \mathbf{z}^{\mathbf{ref}}) \triangleq \bigcup_{i \in 1..|A|} R(\mathbf{z}^i, \mathbf{z}^{\mathbf{ref}}). \tag{5}$$

The $\mathcal{S}$ metric of A with respect to the reference vector is the 'hyperarea', or Lebesgue integral of the set $R(A, \mathbf{z}^{\mathbf{ref}})$. In a minimization (maximization) problem, the reference point $\mathbf{z}^{\mathbf{ref}}$ is taken by Zitzler to be the vector whose components are the maximum (minimum) value in each objective. This gives a non-negative measure for all possible nondominated sets in the feasible objective space[2].

Computing $\mathcal{S}$

The $\mathcal{S}$ metric of a nondominated set A may be computed by recursively projecting the set of points into fewer dimensions and calculating the Lebesgue integral of these. To calculate the $\mathcal{S}$ metric in just two dimensions (objectives) the points are sorted in decreasing order of objective 1 values, and then the following expression is evaluated:

$$\sum_{i \in 1..|A|} |z_1^i - z_1^{ref}| . |z_2^i - z_2^{i-1}| \tag{6}$$

where z_2^0 is initially set to z_2^{ref}. In higher dimensions this generalizes to the recursive function: size_of_set$(A, \mathbf{z}^{ref}, k)$, shown in Figure 6. The function has a time complexity of $O(n^{k+1})$ for general k: the NDk() function is $O(n^2)$ and this must be performed up to n times in a call to size_of_set$(A, \mathbf{z}^{ref}, k)$, not counting further recursive calls; and size_of_set$(A, \mathbf{z}^{ref}, k-1)$ is called n times at level k, for $k \geq 3$. An illustration of the calculation of the $\mathcal{S}$ for three points in three dimensions is given in Figure 7.

[2]N.B. The choice of the reference vector is fairly arbitrary, and could really be any vector beyond the feasible objective space. But different choices will affect the relative $\mathcal{S}$ value of two different nondominated sets (see Figure 5). Although the choice of reference point does affect the ordering of nondominated sets, scaling the objectives does not, i.e., it does not matter if the magnitudes of different objectives are very different.

Archiving strategy based on $\mathcal{S}$ values

We define the Reduce function, S, shown in Figure 8. For archives which are full (i.e. $|M_t| = arcsize$), the resulting archiving algorithm AA_S accepts a vector if it: (a) dominates any member of the archive; or (b) if it is nondominated with respect to (all members of) the archive and its addition would increase the net value of the $\mathcal{S}$ metric [Zitzler, 1999] of the archive when one other (selected) member of the archive is removed to allow its entry.

A proof that this algorithm guarantees convergence to a subset of the Pareto front for bounded archives is given in Appendix A. This result can also be generalized to any quality metric $\mathcal{Q}$ which has the property that $\mathcal{Q}(Z_1) > \mathcal{Q}(Z_2)$ whenever a nondominated set Z_1 weakly outperforms another nondominated set Z_2. Using the $\mathcal{S}$ metric also guarantees that the converged set is a local optimum of $\mathcal{S}$. This means that there is no point which would result in a net increase in the $\mathcal{S}$ metric of the archive if it were added and another point were removed. This guarantee may be important because sets which are local optimum of $\mathcal{S}$ seem to be 'well-distributed'. Unfortunately, at present we have found no way to quantify well-distributed in this context so this observation is not provable. However, see Figure 9 to get some feeling for why a local optimum in $\mathcal{S}$ should be well-distributed in general.

The $\mathcal{S}$ metric archiving has some provable convergence properties and appears to encourage a good distribution of points but its computational overhead is $O(n^{k+1})$. This would be prohibitively high for more than a few objectives, and relatively small archive sizes. Furthermore, the $\mathcal{S}$ metric requires a normalized and positive objective space to work with. In other words, a reference point must be chosen to bound the volume of the nondominated region from above (assuming minimization). The choice of the reference point is rather arbitrary and introduces scaling issues.

4 The Adaptive Grid Archiving Algorithm

In this section we present an adaptive grid archiving (AGA) algorithm that addresses some of the problems with the three archiving strategies considered in the previous section. The algorithm is based on the archiving method used in PAES [Knowles and Corne, 2000]. The AGA algorithm does not have the cycle-preventing property of the $\mathcal{S}$ metric, thus the archive does not converge in the sense given in Definition 2.9. However, several other pseudo-convergence results can be derived, which show that the AGA algorithm will store and maintain a well-distributed and diverse set of vectors.

Figure 10 illustrates the basic principle of AGA. As points in the objective space are generated and archived, the location and size of a grid in the hyper-dimensional space is adapted so that it just envelopes the

points. The grid is used to aid in selecting which points to remove from the archive, should the latter reach its capacity bound. In this case a point from the most crowded region(s) is selected, so long as it is not an extremal point. This strategy ensures archived points cover a wide extent in objective space and are 'well-distributed'.

4.1 Preliminary definitions

We make the following definitions in order to describe the AGA algorithm more formally:

Definition 4.1 *Let the nondominated set of vectors from amongst the archive at time step $t-1$ together with the new vector $\mathbf{z}_t$ generated at time step t be denoted by N_t. That is, $N_t = \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$. Note that $|N_t|$ may be larger than arcsize.*

Definition 4.2 *Let the minimum and maximum scalar values of an objective k amongst the vectors in a vector set Z be denoted $minz_{k,Z}$ and $maxz_{k,Z}$, respectively. That is, $minz_{k,Z} = \min_{\mathbf{z} \in Z}(z_k)$ and $maxz_{k,Z} = \max_{\mathbf{z} \in Z}(z_k)$. The range of the vectors in the set N_t in objective k is given by $range_{k,t} = \max_{z \in N_t}(z_k) - \min_{z \in N_t}(z_k)$.*

Definition 4.3 *There are $2K$* boundaries *of the adaptive grid: $ub_{k,t}$ and $lb_{k,t}$ for all $k \in 1..K$. The boundaries are set so that $\forall t, \forall k, (ub_{k,t} > maxz_{k,N_t}) \wedge (lb_{k,t} < minz_{k,N_t})$. See Figure 11 and Figure 13—Rule:* Update_Boundaries *to see how the boundaries are updated.*

Definition 4.4 *The rectangular polytope defined by the 'corners' $(ub_{1,t}, ub_{2,t}, \ldots, ub_{k,t})$ and $(lb_{1,t}, lb_{2,t}, \ldots, lb_{k,t})$ is divided into a set R_t of similar rectangular polytope* regions *$r_t^{\mathbf{i}} \in R_t$, where $\mathbf{i} = (i_1, i_2, \ldots, i_K)$ is the co-ordinate vector of the region, and $\forall k \in 1..K$, $i_k \in 1..div$ where $div \in \mathbb{Z}, div \geq 2$ is a constant parameter, the number of divisions of the objective space in each dimension, set by the user. The set of all region co-ordinates is I and $|I| = div^K$.*

Definition 4.5 *The boundaries of the regions $r_t^{\mathbf{i}}$ are given by:*
$\forall k \in 1..K$, $rub_{k,\mathbf{i},t} = lb_{k,t} + i_k/div.(ub_{k,t} - lb_{k,t})$ and
$rlb_{k,\mathbf{i},t} = lb_{k,t} + (i_k - 1)/div.(ub_{k,t} - lb_{k,t})$. See Figure 12.

Definition 4.6 *We say that a vector $\mathbf{z}$ in M_{t-1}* occupies *a region $r_t^{\mathbf{i}}$ if $\forall k$, $z_k \geq rlb_{k,\mathbf{i},t} \wedge z_k < rub_{k,\mathbf{i},t}$. We call a region $r_t^{\mathbf{i}}$ that has a vector $\mathbf{z}$ in M_{t-1} occupying it, an* occupied region.

Definition 4.7 *We say that the vector* $\mathbf{z} \in Z^*$ Pareto occupies *a region* $r_t^{\mathbf{i}}$ *if for all* k, $z_k \geq rlb_{k,\mathbf{i},t} \wedge z_k < rub_{k,\mathbf{i},t}$, *even if the vector is not in* M_{t-1}. *We call a region that has a* $\mathbf{z} \in Z^*$ *(and not necessarily in* M_{t-1}*) occupying it, a* Pareto occupied *region (POR).*

Definition 4.8 *The* population $p(r_t^{\mathbf{i}})$ *of a region is the number of vectors in* M_{t-1} *occupying it.*

Definition 4.9 *The set of vectors from a vector set* Z *occupying a region is* $\mathsf{Occ}(r_t^{\mathbf{i}}, Z) \subseteq Z$.

Definition 4.10 *The region occupied by a vector* $\mathbf{z}$ *at time* t *is denoted* $r_t^{\mathbf{i}(\mathbf{z})}$.

If there is no difference between $minz_{k,N_t}$ and $maxz_{k,N_t}$ for some objective k then the adaptive grid boundaries are undefined, and the boundaries of all regions are undefined. In this case, the rules called by $\mathsf{Reduce}()$ that use the grid populations to perform crowding are also undefined. In the following, we make the assumption that the grid boundaries are always well-defined. This is a reasonable assumption since the archive is full whenever any of the rules that use the adaptive grid are executed, so there will normally be some difference between the vectors in M_t in each of the objectives.

We also make the constraint that $arcsize > 2K$. This ensures that the archive is large enough to accommodate all nondominated extremal vectors.

In the adaptive grid archiving algorithm we want to protect uniquely extremal vectors from being removed from the archive once they have entered it (except by domination), so that the vectors in the archive will converge to a set which covers the largest possible range in objective space, in each objective. However, the archiving algorithm will be removing vectors from crowded regions. To avoid removing extremal vectors from these regions we will need a way of counting the number of vectors that occupy a region, (i.e. how crowded it is) *excluding* the number of uniquely extremal vectors. In the following, we define some terms needed to do this.

Definition 4.11 *Let the set of uniquely extremal vectors in* N_t *be defined:*

$$Z_t^{ext} = \{\mathbf{z}^{ext} \in N_t \mid \quad (\exists k \in 1..K, \nexists \mathbf{z} \in N_t, \mathbf{z} \neq \mathbf{z}^{ext}, z_k \leq z_k^{ext}) \vee (\exists k \in 1..K, \nexists \mathbf{z} \in N_t, \mathbf{z} \neq \mathbf{z}^{ext}, z_k \geq z_k^{ext})\}$$

Definition 4.12 *Let the set of non-uniquely extremal (nue) vectors of a region at time* t *be defined:*

$$\mathsf{nue}(r_t^{\mathbf{i}}) = \mathsf{Occ}(r_t^{\mathbf{i}}, Z_t) \setminus Z_t^{ext}$$

Definition 4.13 *Let the population of of non-uniquely extremal vectors of a region at time t be defined:*

$$p_{nue}(r_t^{\mathbf{i}}) = |\mathsf{nue}(r_t^{\mathbf{i}})|$$

Definition 4.14 *Let the set of* crowded regions *be defined:*

$$CR_t = \text{argmax}_{\mathbf{i} \in I}(p_{nue}(r_t^{\mathbf{i}}))$$

Definition 4.15 *Let the set of vectors in the set of crowded regions that are available for removal from the archive at time t be defined:*

$$Z_{c,t} = \bigcup_{r_t^{\mathbf{i}} \in CR_t} \mathsf{nue}(r_t^{\mathbf{i}})$$

In other words, this is the set of vectors that are in the set of most crowded regions, where the most crowded regions are defined as those with the largest population, not counting uniquely extremal vectors. Let the vector $\mathbf{z}^{c,t}$ be a vector selected uniformly at random from within $Z_{c,t}$.

4.2 The algorithm

The adaptive grid archiving algorithm is shown in Figure 13. At each step, the algorithm first updates the boundaries of the adaptive grid in accordance with definitions 4.3–4.5. Then exactly one rule for updating the archive is invoked: Is_Dominated discards $\mathbf{z}_t$ if it is dominated by the archive; Dominates accepts $\mathbf{z}_t$ if it dominates the archive, and discards all dominated members of the archive; Fill accepts $\mathbf{z}_t$ if it is nondominated and the archive is not yet full; Extends accepts $\mathbf{z}_t$ if it increases the extent of the grid in any objective dimension; Reduce_Crowding accepts $\mathbf{z}_t$ if the archive is full but $\mathbf{z}_t$ occupies a less crowded region than some other point in the archive (and discards a point from a crowded region); and Steady_State discards $\mathbf{z}_t$ if the archive is full and the new point cannot be accepted by rules Reduce_Crowding or Dominates.

The overall effect of these rules can be understood as follows. Is_Dominated and Dominates encourage progression towards the true Pareto front. Extends encourages the range of values in objective space represented by the vectors in the archive to be as large as possible. And Reduce_Crowding encourages a uniform distribution of points in the objective space. Clearly, the latter is dependent on the number of grid regions versus the bound on the archive size. Ideally, we would like each member of the archive to occupy its own grid region, to ensure a good distribution of points. Thus, choosing the number of grid regions should be a function of archive size only. In what follows, we show how the number of grid regions can be selected.

The rules of AGA do not constitute an algorithm that converges in the sense of definition 2.9. This is because

nondominated vectors can be removed and cycles of entry and removal of these vectors can ensue. Despite this, it is possible to show that when (if) the grid boundaries converge (i.e. stop moving) a set of regions in the grid will become constantly occupied over time by points in the archive. These grid regions, which we call critical Pareto occupied regions (CPORs), contain the true Pareto front. Thus, we can show that after a time the archive will contain points that are at most a distance of l from the true Pareto front, where l is the large diagonal of a grid region. This constitutes a convergence result for AGA.

4.3 Convergence analysis

In the following we show that the grid regions of AGA can converge under certain conditions.

Lemma 4.1 *If a vector $\mathbf{z}_t \in Z$ with component $z_j = minz_{j,Z}$ for some $j \in 1..K$ is generated at time t, then $minz_{j,M_t} = minz_{j,Z}$.*

Proof 4.1 *If $\mathbf{z}_t$ enters the archive then clearly we have $minz_{j,M_t} = minz_{j,Z}$, as required. On the other hand, if $\mathbf{z}_t$ does not enter the archive, then it must be either (a) dominated by something in the archive, or (b) not outside the old boundaries of the archive. Case (a) means that there must be a vector in the archive which already has $z_j = minz_{j,Z}$, in order to dominate $\mathbf{z}_t$. Case (b) also means that there is a vector in the archive with a component $z_j = minz_{j,Z}$ since otherwise $\mathbf{z}_t$ would have been outside the archive's grid boundaries.*

Lemma 4.2 *If, for some $j \in 1..K, \exists \mathbf{z} \in M_{t_m}$ with $z_j = minz_{j,Z}$ then $\forall t > t_m, \exists \mathbf{z} \in M_t$ with $z_j = minz_{j,Z}$.*

Proof 4.2 *Assume that for some $j \in 1..K, \exists \mathbf{z} \in M_{t_m}$ with $z_j = minz_{j,Z}$. We show that the archiving algorithm is not capable of removing all the vectors with component $z_j = minz_{j,Z_f}$.*

Let us consider each of the rules that can remove vectors. These are the rules Extends, Reduce_Crowding, *and* Dominates.

Dominates *can remove multiple vectors at once. However, it cannot remove any vector with component $z_j = minz_{j,Z}$ except by replacing it with another vector also with a component $z_j = minz_{j,Z}$ since the new vector must dominate the one(s) replaced.*

Extends *can remove only one vector $\mathbf{z}^{c,t}$ from the archive. If there is only one vector $\mathbf{z} \in M_t$ with $z_j = minz_{j,Z}$ then it is a unique extremum in the archive and* Extends *cannot remove it. If there are $n > 1$ vectors in the archive with $z_j = minz_{j,Z}$ then* Extends *may remove one but one or more will remain.*

Reduce_Crowding *can also remove only one vector from the archive and only if it is not a unique extremum. The same argument as for* Extends *applies.*

Theorem 4.1 *The lower boundaries of the grid $lb_{k,t}$ converge for all k.*

Proof 4.3 *To show that the lower boundaries converge it is sufficient to show that there is a time t_m such that $\forall t \geq t_m, \forall k, minz_{k,M_t} = minz_{k,Z_f}$. This is proved by Lemma 4.1 and Lemma 4.2.*

Although it is possible to prove that the lower boundaries converge in all cases, unfortunately this is not true for the upper boundaries. However, it is possible to specify an additional condition under which the upper boundaries can be proved to converge. Section 5 contains proofs showing that the upper boundaries converge under this condition. For now, we introduce this convergence as an assumption:

Assumption 4.1 *The upper boundaries $ub_{k,t}$ converge for all k.*

Corrollary 4.1 *Since the lower boundaries of the grid converge (Theorem 4.1) and by Assumption 4.1, the upper boundaries of the grid also converge, then so do all the boundaries of all of the regions in the grid.*

In the following, our proofs rely on Corollary 4.1. This allow us to consider all of the regions as being static, enabling us to prove that some of the regions containing Pareto optimal points will become constantly occupied by points in the archive. We now introduce some additional terminology relating to the grid regions, needed for these proofs.

Definition 4.16 *Let a set of regions whose boundaries are converged be called a* converged set *of regions, and each region in the set be called a* converged region.

Definition 4.17 *If a converged set of regions also has a subset of regions which remain occupied over time then we say that there is a converged set of occupied regions, R_{COR}. The regions that comprise this set are called* constantly occupied regions *(CORs).*

The following three definitions introduce dominance relationships between regions, and are illustrated by Figure 14.

Definition 4.18 *We say that a region $r^{(1)}$ is* superior *to another region $r^{(2)}$ iff $r^{(1)}$'s coordinates are all strictly less than $r^{(2)}$'s. In addition, $r^{(2)}$ is said to be* inferior *to $r^{(1)}$. Notice that all vectors in $r^{(2)}$ are dominated by vectors in $r^{(1)}$. Thus, any region that is inferior to another Pareto occupied region cannot itself be a Pareto occupied region.*

Definition 4.19 *We say that a region $r^{(1)}$ is* weakly superior *to a region $r^{(2)}$ if $r^{(1)}$'s coordinates dominate $r^{(2)}$'s and $r^{(1)}$ is not superior to $r^{(2)}$. In addition, $r^{(2)}$ is said to be* weakly inferior *to $r^{(1)}$. Notice that it is*

possible for a region that is weakly inferior to another region, to contain efficient vectors, and hence to be a Pareto occupied region.

Definition 4.20 *Two regions are incomparable with respect to each other if neither is superior nor weakly superior to the other.*

Definition 4.21 *In a set of converged regions, Pareto occupied regions that are not weakly inferior to any other Pareto occupied region are called* critical Pareto occupied regions *(CPORs). An important property of a CPOR is the following: no vector that occupies a critical Pareto occupied region can be dominated by any vector $z \in Z$ that occupies any other region. The set of CPORs is denoted R_{CPOR}.*

Definition 4.22 *A Pareto non-inferior region PNIR is any (converged) region that is weakly inferior to a Pareto occupied region, or is itself a Pareto occupied region. Figure 16 illustrates the concept of a PNIR.*

Note that the set of Pareto occupied regions (PORs) contains the Pareto front. The set of $PNIR$s includes these, together with any additional regions (such as $r^{2,5}$ in Figure 16) which, informally, fill in the gaps. That is, given any path $(r^{(1)}, r^{(2)}, r^{(3)}, r^{(4)}, \ldots, r^{(n)})$ where $r^{(j+1)}$ shares at least one corner with $r^{(j)}$, and in which $r^{(1)}$ is inferior to a POR and $r^{(n)}$ is superior to a POR, the path must include at least one $PNIR$.

Lemma 4.3 *If a critical Pareto occupied region is occupied at time t_i, then it is occupied for all $t > t_i$.*

Proof 4.4 *Assume at time t_i a CPOR has a population of $n \geq 1$ vectors $\mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3, \ldots, \mathbf{z}^n$, and at some time $t_j > t_i$ all of them are removed from the archive.*

Only Rules Reduce_Crowding *and* Dominates *can remove vectors from the archive*[3]*. However,* Reduce_Crowding *can only remove one vector from a region. Therefore, if $n > 1$ it cannot remove all the vectors.*

On the other hand, if $n = 1$, then Reduce_Crowding *cannot remove the single vector in the CPOR because $p_{nue}(r_t^{\mathbf{i}(CPOR)}) \not> 1$. That is, the critical Pareto occupied region we are considering does not have a population (of non-uniquely extremal vectors) greater than 1. Therefore, either it is not one of the most crowded regions i.e. it is not in CR_t or there does not exist any region with a population greater than 1. In either case,* Reduce_Crowding *cannot remove a vector from this critical Pareto occupied region.*

Thus, Rule Dominates *must remove the n vectors. But, by the definition of a critical Pareto occupied region, no vectors exist that can dominate any vector in a CPOR, except another vector in the same CPOR. Therefore* Dominates *can remove the n vectors only by replacing them with another vector in the same CPOR.*

[3]Extends cannot execute because we have already assumed that the region boundaries have converged.

Therefore all cases contradict our original assumption that there is a time $t_j > t_i$ such that no vectors in the archive occupy the $CPOR$.

Theorem 4.2 *The maximum number of mutually non-inferior regions in a K dimensional vector space divided up into div equal divisions in each dimension, is $div^K - (div-1)^K$. See Figure 19.*

Proof 4.5 *The proof of this theorem is given in Appendix B.*

Lemma 4.4 *For all t, if $arcsize > div^K - (div-1)^K + 2K$ and an efficient point $\mathbf{z}_t \in Z^*$ is generated by* Gen(t)*, then $p(r_t^{\mathbf{i}(\mathbf{z}_t)}) \geq 1$. In other words, if a Pareto optimal point is generated at time t, its region will be occupied in the same time step, provided the archive capacity is greater than $div^K - (div-1)^K + 2K$.*

Proof 4.6 *Consider the time t when $\mathbf{z}_t \in Z^*$ is generated by* Gen(t)*. Then exactly one of* Fill, Dominates, Reduce_Crowding *or* Steady_State *executes. (*Is_Dominated *cannot execute since $\mathbf{z}_t$ is Pareto optimal and* Extends *cannot execute because we have already assumed that the grid boundaries have converged.)*

If either Fill, Dominates *or* Reduce_Crowding *executes then $\mathbf{z}_t$ is accepted and clearly $p(r_t^{\mathbf{i}(\mathbf{z}_t)}) \geq 1$.*

Obviously, if $\mathbf{z}_t$ dominates any member of M_{t-1}, then Dominates *will execute. And if $|M_{t-1}| < arcsize$ then* Fill *will execute.*

But if $|M_{t-1}| = arcsize$ and $\mathbf{z}_t$ does not dominate M_{t-1} then either Steady_State *or* Reduce_Crowding *can execute. We must now show that* Steady_State *does not execute when $p(r_{t-1}^{\mathbf{i}(\mathbf{z}_t)}) = 0$ i.e. it can only execute when $\mathbf{z}_t$'s region is already occupied, which would leave it occupied and we would still have $p(r_t^{\mathbf{i}(\mathbf{z}_t)}) \geq 1$.*

So we have $\mathbf{z}_t \sim M_{t-1}$ and $|M_{t-1}| = arcsize$. The latter implies $|M_{t-1}| > div^K - (div-1)^K + 2K$. From Theorem 4.2, and the fact that a set of mutually nondominated vectors must occupy a set of mutually non-inferior regions (see Figure 17), it follows that M_t occupies at most $div^K - (div-1)^K$ regions. Now, at most $2K$ vectors are uniquely extremal. Therefore, it follows that $\exists \mathbf{i}, p_{nue}(r_t^{\mathbf{i}}) > 1$. Now, if $p(r_{t-1}^{\mathbf{i}(\mathbf{z}_t)}) = 0$ then $\exists \mathbf{i}, p(r_{t-1}^{\mathbf{i}}) > p(r_{t-1}^{\mathbf{i}(\mathbf{z}_t)})$. So, we now have $\exists \mathbf{i}, p_{nue}(r_t^{\mathbf{i}}) > 1 \wedge \exists \mathbf{i}, p(r_{t-1}^{\mathbf{i}}) > p(r_{t-1}^{\mathbf{i}(\mathbf{z}_t)})$, therefore Reduce_Crowding *executes.*

Theorem 4.3 *$\exists t_m$ such that $\forall t > t_m, \forall r_t^{\mathbf{i}} \in R_{CPOR}, p(r_t^{\mathbf{i}}) > 0$, provided $arcsize > div^K - (div-1)^K + 2K$. In other words, there is a time after which all of the critical Pareto occupied regions become constantly occupied, provided the archive is sized appropriately.*

Proof 4.7 *From lemmata 4.3 and 4.4 we see that the number of critical Pareto occupied regions that are occupied by vectors in the archive monotonically increases over time, provided $arcsize > div^K - (div -$*

$1)^K + 2K$. *Since the number of critical Pareto occupied regions is finite this implies that the set of occupied* $CPORs$ *converges to* R_{CPOR}.

Theorem 4.4 *If at time* t_m *the set of occupied* $CPORs$ *has converged to* R_{CPOR} *then for all* $t > t_m$, *all vectors in* M_t *reside in a* $PNIR$.

Proof 4.8 *Regions that are not* $PNIRs$ *are either weakly superior to, superior to, or inferior to, a critical Pareto occupied region. But for all* t, *no vector generated by* Gen(t) *can lie in a region that is superior or weakly superior to any critical Pareto occupied region, since they are all occupied for all future* t. *And, any vector generated by* Gen(t) *that is inferior to an occupied region will not be accepted — from the definition of domination and the rule* Is_Dominated.

The implications of Theorems 4.3 and 4.4 are that the vectors in the archive will become well distributed and close to the Pareto front in a well-defined way: after a certain number of iterations, a subset of them will always occupy all of the critical Pareto occupied regions, and the remainder will be in a $PNIR$. In order to guarantee achieving this, the archive should be given a capacity of $1 + div^K - (div - 1)^K + 2K$, at least.

In practice, however, the archive can probably be given a capacity smaller than this, provided it is larger than the number of critical Pareto occupied regions. We made the constraint that $arcsize > div^K - (div-1)^K + 2K$ in order to prove Lemma 4.4, which ensures that if an efficient point is generated, its region becomes or remains occupied. This was needed to prove that all $CPOR$s will become constantly occupied. But we would generally expect all of the $CPOR$s would become (constantly) occupied even without Lemma 4.4. This is because, as a result of rule Dominates, non-efficient vectors will generally be removed in favour of efficient ones that dominate them, over time. Thus, any set of mutually nondominated vectors that do not occupy all of the critical Pareto occupied regions, will *usually* give way to an archive in which one more critical Pareto occupied region is occupied. The reason this is not *guaranteed* is because it may be the case that all of the unfound vectors in the critical Pareto occupied regions do not dominate any vector in the archive, for all future iterations. Assuming this unlikely event does not occur, there will still be a monotonic increase in the occupation of critical Pareto occupied regions. Thus, if the archive is greater than the number of critical Pareto occupied regions then eventually all of the critical Pareto occupied regions will become constantly occupied.

So, if a small chance of "sub-optimal" convergence can be tolerated then the requirement for $arcsize$ goes down to the maximum number of critical Pareto occupied regions (plus the number of extremal Pareto occupied regions) that there are in a grid space.

Conjecture 4.1 *The maximum number of critical Pareto occupied regions in an objective space of K dimensions and divided into div divisions in each dimension is given by:*

$$v_s = \sum_{j=0}^{(s-K)/div} \left[(-1)^j . \binom{K}{j} . \binom{s - div.j - 1}{K-1} \right] \tag{7}$$

when s is set to $\lfloor K/2(div+1) \rfloor$.

A proof of this conjecture is not provided but some supporting argument is given. A critical Pareto occupied region ($CPOR$) cannot have coordinates that put it on the same one dimensional 'row' as any other $CPOR$, otherwise one of them would be weakly superior to the other. From this we can see that in any list of critical Pareto optimal regions, no pair of regions can have an identical set of any $K-1$ coordinates. In addition, all the regions must be non-inferior so no two vectors should share the same root region. Such an arrangement of regions can be achieved if the sum of all the coordinates of each region is equal to a constant value. This ensures that the above constraints are satisfied because any two regions that are different will be different in $K-1$ coordinates if their sum is equal, and any two regions with the same root region would surely have a different coordinate sum. The number of regions that have the same coordinate sum is given by equation 7. This equation actually gives the number of ways there are of achieving a total of s from K dice, each with div faces. The equation was obtained from [Rob, 2001], and a derivation is given in Appendix C When the sum is set to $\lfloor K/2(div+1) \rfloor$ this number appears to be maximized, although no proof of this has yet been found.

4.4 Summary of the AGA algorithm's properties

In summary, the adaptive grid archiving algorithm is guaranteed to:

- store and maintain points at the extremes of all objectives (Theorems 4.1 and 5.1);
- store and maintain points in all of the critical Pareto occupied regions (Theorem 4.3);
- and distribute its remaining points amongst the Pareto non-inferior regions (Theorem 4.4),

provided the upper boundaries of the grid converge, $arcsize > div^K - (div-1)^K + 2K$, the generating function generates all points with positive probability, and the search space is finite.

Table 1 gives the required size of the archive for different values of div and K, for the case where convergence is *guaranteed* (3rd column), and the case where it is only probable (4th column). Clearly, in the two-dimensional case, the required archive size to guarantee convergence is small, even with 64 divisions per objective. As

the number of objectives increases, the required archive sizes for guaranteed convergence rapidly increase to numbers that are larger than usually used as the population in an EA. However, with 8 objectives and 4 divisions per objective, $arcsize > 8092$, a manageable number, is large enough to cover the $CPOR$s, if our conjecture 4.1 is true.

4.5 Computational cost of AGA

The computational overhead of using the adaptive grid archiving algorithm can be made quite low if the grid is implemented as a quadtree (as in PAES [Knowles and Corne, 2000]). In this way the grid region of a new point can be identified in the logarithm of the number of grid regions. With simple bookkeeping of the populations of each grid region it is then relatively cheap to determine the update of the archive.

5 Convergence Problems With Adaptive Grid Methods

The convergence proofs given in the last section were based on assuming that the upper grid boundaries of the adaptive grid eventually converge. In this section, we show under which conditions this assumption holds, and show that special cases of this are when there are only two objectives, and when the Pareto front is as broad as the entire search space (in all objective dimensions). More generally, however the assumption does not hold. To show the latter we begin by giving an explicit example in which the upper grid boundaries will never converge. This example shows that, in AGA, cycles of entry and removal of points can prevent the boundaries from ever converging. In the next section we compare this drawback of the adaptive grid algorithm to a related behaviour exhibited by ϵ-Pareto archiving algorithms.

Consider a three-objective vector space in which the vector in the efficient set with largest value in objective 1 is $\mathbf{z}^1 = (6, 1, 3)$. Another vector in the Pareto set is $\mathbf{z}^2 = (6, 3, 1)$. However, the non-efficient vector $\mathbf{z}^3 = (7, 3, 2)$ also exists. If $\mathbf{z}^1$ (but not $\mathbf{z}^2$) is currently in the archive and $\mathbf{z}^3$ is generated at the next step, then it will be accepted and the grid boundaries will be updated. Thus, the upper grid boundary on objective 1 will become larger than any vector in the efficient set's value in this objective. However, this value of the grid boundary may not be maintained because if $\mathbf{z}^2$ is generated at some future time, $\mathbf{z}^3$ will be dominated, and removed from the archive, and the grid boundaries will be appropriately updated, accordingly. Because it is possible for $\mathbf{z}^2$ to be 'lost' from the archive, in future iterations, the same cycle of events can occur again. Therefore, in this kind of situation the upper grid boundaries may never converge.

Condition A.1, below, formalizes the minimal properties which must be true of Z^* if we are to avoid situations such as the above, hence enabling us to prove that the upper grid boundaries will converge. In the following,

we prove the convergence of the upper boundaries in the restricted case when Condition A.1 is true.

Condition 5.1 *$\forall k, \nexists \mathbf{z} \in C(Z^*), z_k \geq maxz_{k,Z^*}, \exists \mathbf{z}^* \in Z^*, z_k^* = maxz_{k,Z^*}, \mathbf{z} \sim \mathbf{z}^*$, where $C(Z^*)$ denotes the complement of the efficient set. In other words, on any objective, there are no non-efficient vectors that have a component value that is larger than or equal to the largest component value of any efficient vector, and that are also nondominated with respect to any of the efficient vectors with the maximum component value.*

Lemma 5.1 *If Condition 5.1 is true and a vector $\mathbf{z}^*$ with component $z_k = maxz_{k,Z^*}$ for some k is generated at time t, then $maxz_{k,M_t} = maxz_{k,Z^*}$.*

Proof 5.1 *Assume that a vector $\mathbf{z}^*$ with component $z_k = maxz_{k,Z^*}$ for some k is generated at time t. Then we have $maxz_{k,N_t} = maxz_{k,Z^*}$.*

The vector $\mathbf{z}^*$ will enter the archive if Dominates *or* Extends *or* Reduce_Crowding *or* Fill(t) *executes, and hence it is clear that $maxz_{k,M_t} \geq maxz_{k,Z^*}$. But because Condition 5.1 is true, we also know that M_t, which is always a nondominated set, cannot contain any non-efficient vector with $z_k \geq maxz_{k,Z^*}$. Thus, $maxz_{k,M_t} = maxz_{k,Z^*}$.*

if $\mathbf{z}^*$ does not enter the archive, it is because Steady_State *executes. However, one of the conditions for* Steady_State *to execute is that $\forall k, maxz_{k,N_t} = maxz_{k,M_{t-1}}$. So we have $z_k = maxz_{k,Z^*}$ and $\forall k, maxz_{k,N_t} = maxz_{k,M_{t-1}}$, and* Steady_State *executes. Therefore, since $\mathbf{z}^* \in N_t$, then $maxz_{k,M_{t-1}} = maxz_{k,N_t} = maxz_{k,Z^*}$, and because* Steady_State *executes $maxz_{k,M_t} = maxz_{k,M_{t-1}}$, so $maxz_{k,M_t} = maxz_{k,Z^*}$, as required.*

Lemma 5.2 *If Condition 5.1 is true, and for some $k \in 1..K, \exists \mathbf{z}^* \in M_{t_m}$ with $z_k = maxz_{k,Z^*}$ then $\forall t > t_m, \exists \mathbf{z} \in M_t$ with $z_k = maxz_{k,Z^*}$.*

Proof 5.2 *Assume that for some $k \in 1..K, \exists \mathbf{z}^* \in M_{t_m}$ with $z_k = maxz_{k,Z^*}$. We show that no archiving rule is capable of removing all the vectors with component $z_k = maxz_{k,Z^*}$.*

Let us consider each of the rules that can remove vectors. These are the rules Extends, Reduce_Crowding, *and* Dominates.

Extends *can remove only one vector from the archive. If there is only one vector $\mathbf{z} \in M_t$ with $z_k = maxz_{k,Z^*}$ then, since Condition 5.1 is true, there can be no vectors in M_t with $z_k > maxz_{k,Z^*}$ (because they would be dominated). Thus $\mathbf{z}$ is a unique extremum in the archive and* Extends *cannot remove it by the definition of* Extends.

On the other hand, if there are $n > 1$ vectors in the archive with $z_k = maxz_{k,Z^*}$ then Extends *may remove one of them. However, one or more will remain in the archive.*

Reduce_Crowding *can also remove only one vector from the archive and only if it is not a unique extremum. Thus, the same argument as for* Extends *applies, so* Reduce_Crowding *cannot remove all vectors with $z_k = maxz_{k,Z^*}$.*

Dominates *can remove multiple vectors at once. However, it cannot remove any efficient vector. Since Condition 5.1 is true any vector in M_t with component $z_k = maxz_{k,Z^*}$ must be efficient. Therefore* Dominates *cannot remove these vectors.*

Theorem 5.1 *If Condition 5.1 is true then the upper boundaries of the grid $ub_{k,t}$ converge for all k.*

Proof 5.3 *To show that the upper boundaries converge it is sufficient to show that there is a time t_m such that $\forall t \geq t_m, \forall k, maxz_{k,M_t} = maxz_{k,Z^*}$. For this we just require:*

1. *If a vector $\mathbf{z}^*$ with component $z_k = maxz_{k,Z^*}$ for some k is generated at time t, then $maxz_{k,M_t} = maxz_{k,Z^*}$. This is Lemma 5.1.*
2. *If for some $k \in 1..K, \exists \mathbf{z}^* \in M_{t_m}$ with $z_k = maxz_{k,Z^*}$ then $\forall t > t_m, \exists \mathbf{z} \in M_t$ with $z_k = maxz_{k,Z^*}$. This is Lemma 5.2.*

Due to Condition 5.1, Item 2. ensures that $\forall t \geq t_m, \forall k, maxz_{k,M_t} = maxz_{k,Z^*}$ because vectors with $z_k > maxz_{k,Z^*}$ are always dominated.

We now provide two cases when Condition 5.1 is always true, so that the upper boundaries of the adaptive grid converge.

Theorem 5.2 *if $K = 2$ then Condition 5.1 is true.*

Proof 5.4 *Assume $K = 2$ and Condition 5.1 is not true. Then there is a vector $\mathbf{z} \in C(Z^*)$ and an objective $k \in 1..2$ such that $z_k \geq maxz_{k,Z^*}$, and $\mathbf{z}$ is nondominated with respect to a vector $\mathbf{z}^* \in Z^*$ with component $z_k = maxz_{k,Z^*}$.*

Without loss of generality, we choose $k = 1$. Since $z_1 \geq maxz_{1,Z^*}$ and $\mathbf{z} \sim \mathbf{z}^*$, then we have $z_2 < z_2^*$. But this must mean that $\mathbf{z}$ is efficient (a contradiction) because if it were not there would have to be another vector $\mathbf{z}^{**} \in Z^* < \mathbf{z}$. But then $\mathbf{z}^{**} < \mathbf{z}^*$ because $z_1^{**} \leq maxz_{k,Z^*}$ and $z_2^{**} \leq z_2$. This is also a contradiction since $\mathbf{z}^*$ is efficient.

Theorem 5.3 *if* $\forall k, \nexists \mathbf{z} \in C(Z^*), z_k \geq maxz_{k,Z^*}$ *then Condition 5.1 is true. In other words, if the efficient set spans the feasible objective space in all objectives then Condition 5.1 is true.*

Proof 5.5 $\forall k, \nexists \mathbf{z} \in C(Z^*), z_k \geq maxz_{k,Z^*} \Rightarrow \forall k, \nexists \mathbf{z} \in C(Z^*), z_k \geq maxz_{k,Z^*}, \exists \mathbf{z}^* \in Z^*, z_k^* = maxz_{k,Z^*}, \mathbf{z} \sim \mathbf{z}^*$.

6 Archiving Algorithms Presented in Laumanns *et al.* (2001)

In a recent technical report [Laumanns et al., 2001], a number of algorithms for maintaining a nondominated solutions archive based on the principle of ϵ-dominance were proposed. Briefly, a point $\mathbf{z}^1$ ϵ-dominates a point $\mathbf{z}^2$ iff $(1+\epsilon).z_k^1 \geq z_k^2$ for all $k \in 1..K$, and for some $\epsilon > 0$. (Here we assume maximization).

6.1 Maintaining an ϵ-approximate Pareto set

In the first algorithm put forward, a new point $\mathbf{z}$ is accepted into the archive iff there is not a point already in the archive which ϵ-dominates $\mathbf{z}$, for a given fixed ϵ. If $\mathbf{z}$ is accepted, all dominated vectors are subsequently removed from the archive so that it remains a mutually nondominated set. This is just like unbounded archiving (section 3.1), except that the criterion for accepting a new point is its relationship with existing points in the sense of e-dominance, rather than ordinary dominance.

Such an algorithm has several desirable properties. First, the size of the archive is bounded by a function of the range of values in the objective space, and the value of ϵ. Second, for each point in a sequence of points presented to the archive, the archive contains a point which ϵ-dominates it. Another way of saying this is that the archive is always an ϵ-approximate Pareto set of the set of points in the sequence presented to it. This property ensures that the points stored in the archive are a diverse, representative subset of the points in the sequence. Third, the algorithm is very simple and has low computational cost.

6.2 Maintaining an ϵ-Pareto set

The second archiving algorithm presented generates an ϵ-Pareto set, that is, a set which is both ϵ-approximate and contains only Pareto optimal points from the sequence of points presented to the archive. When a new point $\mathbf{z}$ is generated it is assigned a "box". The box has a vector index which is a function of the point's position in objective space and ϵ, with some rounding, so that there is a many-to-one mapping between

points in objective space and boxes. Specifically the box index vector $b = (b_1, \ldots b_K)$ is calculated using:

$$b_k = \lfloor \frac{logz_k}{log(1+\epsilon)} \rfloor \tag{8}$$

for all $k \in 1..K$.

A point $\mathbf{z}$ is accepted into the archive if it's box dominates the box of any point in the archive. Subsequently all vectors in these dominated boxes are removed. A point $\mathbf{z}$ may also be accepted if it is in the same box as a point $\mathbf{z}'$ in the archive, and $\mathbf{z}$ dominates $\mathbf{z}'$. Finally, $\mathbf{z}$ may be accepted if it is in a box which is both new (not in the archive) and not dominated by any box in the archive.

Laumanns *et al.* show that this archiving algorithm also produces an archive of bounded size for a given fixed value of ϵ.

6.3 Dynamic adaptation of ϵ

In both of the above schemes the maximum size of the archive cannot be set *a priori* without knowledge of the ranges of values of the points in objective space. Thus if ϵ is set too large or too small, the size of the archive may be correspondingly too large or small with respect to the desired number of solutions. To overcome this problem, Laumanns *et al.* propose methods for dynamically adapting the value of ϵ so that the size of the archive never exceeds a predetermined bound.

6.3.1 The ϵ-approximate case

In the first method, which is designed to work with the first algorithm described above for producing an ϵ-approximate set, ϵ is initially set as a function of the range in objective space of the first pair of mutually nondominated points, and the required bound on archive size. Subsequently, if the bound will be exceeded a new ϵ is calculated using the new ranges of the points in the archive. Using the new ϵ, all archive members are again compared in the order they appeared in the sequence presented to the archiving algorithm. Whenever one archive member is ϵ-dominated by an older one, the younger is removed.

Note that ϵ can only increase with time in this proposed method. Thus if the Pareto front of solutions gets smaller at the end of a run, after being initially large, then the archive may become a poor approximation to it, because with large ϵ, new, better solutions cannot join the archive because they are ϵ-dominated by members of the archive which are not nearly as good.

A further problem is that whenever ϵ is increased one must scan through the entire archive in sequence

order, removing elements which are ϵ-dominated by older elements. Obviously, this is an intensive operation, requiring $O(|A|^2)$ steps for an archive size of $|A|$.

6.3.2 The ϵ-Pareto case

The methods proposed in [Laumanns et al., 2001] for obtaining a fixed size ϵ-Pareto set are only sketched there, and no proofs are given for their convergence properties. Nonetheless, the authors propose that the box-based update rule be used again, and as above, ϵ, starting from a minimal value, is only ever increased with time. Boxes are then joined together, to accommodate the increases in ϵ. Laumanns *et al.* states without proof that in the worst case, this method results in a fixed cardinality ϵ-Pareto Set, which ϵ-dominates an area at most twice that of the size of the actual Pareto set.

Nonetheless, Laumanns *et al.* point out that when the objective ranges of the Pareto front are much smaller than the ranges of the points generated in the sequence then ϵ may become so large that in the end only one point in the archive is found; all other Pareto points being ϵ-dominated by this one point.

To counteract this problem, Laumanns *et al.* suggest the use of a two-stage approach in which reasonable bounds are calculated in a first run, and in a second run a fixed value of ϵ is used. This seems rather inelegant but provided two runs can be afforded it may be an acceptable alternative which avoids the problems with the dynamic schemes.

6.4 Comparison with Adaptive Grid Archiving

With the methods proposed by Laumanns *et al.* described above one must either pre-set the value of ϵ, or bound the size of the archive and use an adaptive setting of ϵ. In the former case, the number of points in the archive is bounded only by some function of the (unknown) objective space ranges. Whereas in the latter case ϵ may become arbitrarily large and so the final set achieved may be a poor representation of the sequence of points presented to the archiving algorithm. Specifically, the number of points in the final archive may be far *fewer* than were desired. However, in either case convergence *is* guaranteed.

In some sense, our adaptive grid archiving (AGA) algorithm complements this tradeoff. In AGA, convergence is not guaranteed in the general case because the upper boundaries of the grid may fluctuate. However, the grid adapts itself to the ranges of points in the archive so that even if the Pareto front gets smaller over time, the archive will generally contain the desired number of solutions, and they will tend to be distributed among the different grid regions. It remains to be seen whether, in practical applications, the potential fluctuation of the grid boundaries is really problematic.

Furthermore, in AGA, if the ranges of the grid *were* set in advance, that is, no adaptive scheme were used, and the archive bound and number of grid regions were set appropriately (as described above), then a result similar to that of Laumanns' algorithm for the ϵ-approximate set is obtained. That is, for every point generated in a sequence presented to the archiving algorithm, we would have a point in the archive which is 'close by'. This closeness can be specified exactly using the following fact which we state here without proof: if all the points in the $CPOR$s are moved to the minimal coordinates of their region (assuming minimization) then they dominate all points generated in the sequence. Thus, the maximum distance that a point in the archive must be moved in order to dominate all points in the sequence presented to the archiving algorithm is the length of the large diagonal in a rectangular polytope grid region. This is a similar convergence result to that of ϵ-approximation.

7 Summary, Conclusions and Further Work

In this paper we have analysed the state-of-the-art in archiving algorithms for use in MOEAs and other multiobjective search algorithms. We began with an analysis of three simple, convergent archiving strategies, noting their strengths and weaknesses. Following this, an adaptive grid archiving algorithm was proposed which has low computational cost, adapts itself to the ranges of values of points in the objective space, and maintains a nondominated-set archive which uses 'crowding' to encourage an even distribution of points. For this algorithm we showed that although convergence to a subset of the true Pareto front is not guaranteed, under certain conditions the grid boundaries do converge. When this occurs, certain grid regions will become constantly occupied, guaranteeing a certain minimum quality of points in the archive, and encouraging diversity.

We also noted the conditions under which the AGA's upper grid boundaries do not converge. Loosely speaking this occurs when the Pareto front has a smaller extent than the whole objective space. This causes a problem because points not in the true Pareto front may cause the grid ranges to be extended, then at some later time these must be reduced again when the points are removed because they become dominated. A similar problem occurs in the adaptive methods proposed by Laumanns *et al.* There, in order to guarantee convergence, ϵ cannot be reduced over time and thus if the final Pareto front is small compared to the ranges of all the points that have been encountered, then ϵ may be too large resulting in a poor representation of the Pareto front. In the worst case, ϵ may have become so large as to lead to an archive of just one point which ϵ-dominates the whole Pareto front.

Thus, in adaptive schemes, there is a conflict between the desire to find a given number of solutions (no more and no fewer), and the need to have guaranteed convergence to a set which closely approximates the entire

Pareto front of the points encountered. The adaptive grid archiving algorithm presented here offers a different compromise to this conflict than the schemes proposed by Laumanns *et al.* Further work should attempt to empirically test and compare the performance of these different archiving algorithms on both real sequences of points found through search, and also worst-case scenarios. From this, we may understand better which schemes are most appropriate for a given problem and desired outcome. We may also understand how to make improvements to these methods.

References

[Hanne, 1999] Hanne, T. (1999). On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research*, 117(3):553–564.

[Hansen and Jaszkiewicz, 1998] Hansen, M. P. and Jaszkiewicz, A. (1998). Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Computing Science, Poznan University of Technology, ul. Piotrow 3a, 60-965 Poznan, Poland.

[Horn, 1997] Horn, J. (1997). Multicriterion decision making. In Bäck, T., Fogel, D., and Michalewicz, Z., editors, *Handbook of Evolutionary Computation*, volume 1, pages F1.9:1 – F1.9:15. IOP Publishing Ltd. and Oxford University Press.

[Knowles and Corne, 2000] Knowles, J. D. and Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172.

[Laumanns et al., 2001] Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2001). 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, Gloriastrasse 35, CH-8092 Zurich, Switzerland.

[Rob, 2001] Rob (2001). Probability of a sum on multiple dice. URL: www.drmath.com/dr.math/problems/regan.3.26.01.html.

[Rudolph and Agapie, 2000] Rudolph, G. and Agapie, A. (2000). Convergence properties of some multi-objective evolutionary algorithms. In *Proceedings of the 2000 Conference on Evolutionary Computation*, volume 2, pages 1010–1016, Piscataway, New Jersey. IEEE Press.

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

A Convergence Proof for $\mathcal{S}$ metric archiving

Let the $\mathcal{S}$ metric of the set M_t be denoted $\mathcal{S}(M_t)$. We note that the $\mathcal{S}$ metric has the following properties necessary for the proofs that follow:

- $\forall M_{t_n} = M_{t_m}(\mathcal{S}(M_{t_n}) = \mathcal{S}(M_{t_m}))$ if the upper bounding values of the dominated region are constant over time.
- The value of the $\mathcal{S}$ metric is maximal for Z^* so the value of the $\mathcal{S}$ metric is bounded.
- The value of the $\mathcal{S}$ metric is strictly greater for M_{t_j} than M_{t_i} if $M_{t_j}\ O_W\ M_{t_i}$.

Lemma A.1 *If M_{t-1} is a nondominated set and the rule* Size *is executed at time t then M_t is also a nondominated set. In other words, rule* Size *maintains the archive as a nondominated set.*

Proof A.1 *Assume M_{t-1} is a nondominated set and the rule* Size *is executed at time t and M_t is* not *a nondominated set. Since rule* Size *is executed, $\mathbf{z}_t$ cannot dominate M_{t-1}. Therefore, we have $\mathbf{z}_t > M_{t-1} \vee \mathbf{z}_t \in M_{t-1} \vee \mathbf{z}_t\ M_{t-1}$. If $\mathbf{z}_t > M_{t-1} \vee \mathbf{z}_t \in M_{t-1}$ then $\mathbf{z}_t$'s addition would not increase the $\mathcal{S}$ value of the archive, contradicting the assumption that rule* Size *is executed. So, $\mathbf{z}_t\ M_{t-1}$. When rule* Size *executes, $M_t = M_{t-1} \cup \mathbf{z}_t \setminus \{\mathbf{z}^{min} \in M_{t-1}\}$. Since $\mathbf{z}_t\ M_{t-1}$, $M_{t-1} \cup \mathbf{z}_t$ is a nondominated set. Therefore M_t is a nondominated set since the removal of any vector from a nondominated set leaves a nondominated set. This contradicts our assumption that M_t is* not *a nondominated set.*

Lemma A.1 shows that, as before, the full set of rules in AA_S ensures that the archive is always a nondominated set. We now prove some convergence properties of AA_S.

Lemma A.2 *M_t is converged under algorithm AA_S implies M_t is a subset of Z^*.*

Proof A.2 *Assume at some time $t = t_i$, M_{t_i} is converged and $M_{t_i} \not\subseteq Z^*$. The latter implies that there exists at least one efficient vector $\mathbf{z}^* \in Z^*, \mathbf{z}^* < M_{t_i}$. We wish to show that at some future time $t_j > t_i$, $M_{t_j} \neq M_{t_i}$.*

We may choose the time $t = t_j$ when Gen(t) *generates the vector $\mathbf{z}^*$ which dominates the archive. Now if $M_{t_j-1} \neq M_{t_i}$ then M_t was not converged, a contradiction to our assumption. Otherwise, by archiving rule* Dominates, *$\mathbf{z}^*$ will replace those vectors in M_{t_j-1} that it dominates. This also contradicts our original assumption that the set M_{t_i} is converged, and we must conclude that vector sets that are not subsets of Z^* are not converged, under AA_S.*

Lemma A.3 *$M_{t_n} \neq M_{t_m}$ for $t_n > t_m$ implies $\forall t > t_n (M_t \neq M_{t_m})$.*

Proof A.3 *Assume $M_{t_n} \neq M_{t_m}$ and $\exists t_p$ such that $M_{t_p} = M_{t_m}$ with $t_p > t_n > t_m$. Since rules* Size, Fill(t), *and* Dominates *all strictly increase $\mathcal{S}(M)$, and rule* Steady_State *leaves $\mathcal{S}(M)$ unchanged, $\mathcal{S}(M_{t_n}) > \mathcal{S}(M_{t_m})$. By the same token, $\mathcal{S}(M_{t_p}) \geq \mathcal{S}(M_{t_n})$ thus $\mathcal{S}(M_{t_p}) > \mathcal{S}(M_{t_m})$, therefore $M_{t_p} \neq M_{t_m}$, a contradiction.*

Lemma A.4 *M_t converges under algorithm $AA_{\mathcal{S}}$.*

Proof A.4 *Assume M_t never converges i.e. $\forall t_i, \exists t_j,\ t_j > t_i,\ M_{t_j} \neq M_{t_i}$. This implies that there are an infinite number of different sets M_t, since none can be revisited (lemma A.3). However, since the set Z is finite, so is 2^Z. But all $M \in 2^Z$ so there are finite different M.*

Theorem A.1 *M_t converges to a subset of Z^* under algorithm $AA_{\mathcal{S}}$.*

Proof A.5 *From lemma A.2 we see that all converged sets are Pareto optimal subsets. From lemma A.4 we see that M_t converges.*

Note A.1 *Theorem A.1 also applies with any metric $\mathcal{Q}$ in place of $\mathcal{S}$, so long as Q is bounded and $\forall M_{t_n} = M_{t_m} (\mathcal{Q}(M_{t_n}) = \mathcal{Q}(M_{t_m}))$ and $\mathcal{Q}(M_{t_j}) > \mathcal{Q}(M_{t_i})$ if $M_{t_j}\ O_W\ M_{t_i}$ since only these properties of the $\mathcal{S}$ metric were used to prove Theorem A.1.*

B Proof of Theorem 4.2

Conjecture B.1 *The maximum number of mutually non-inferior regions in a K dimensional vector space divided up into div equal divisions in each dimension, is $div^K - (div-1)^K$.*

Lemma B.1 *There are $div^K - (div-1)^K$ regions with a co-ordinate vector $\mathbf{c} = (c_1, c_2, \ldots, c_K)$ such that $\forall i \in 1..K, c_i = 1..div, \exists j \in 1..K, c_j = 1$. In other words, there are $div^K - (div-1)^K$ regions with a co-ordinate vector in which at least one of component has the value 1. See Figure 19*

Proof B.1 *Let $\mathbf{c} = (c_1, c_2, \ldots, c_K)$ be the co-ordinate of a region, with $\forall i \in 1..K, c_i \in 1..div$. Clearly there are div^K such regions.*
Let $\mathbf{d} = (d_1, d_2, \ldots, d_K)$ be the co-ordinates of a region with no components having the value 1. So, $\forall i \in 1..K, d_i \in 2..div$. Clearly there are $(div-1)^K$ of these regions. Therefore, by subtracting the second set of regions from the first, we have that there are $div^K - (div-1)^K$ regions with a co-ordinate vector in which at least one of its components has the value 1.

Definition B.1 *Every region with co-ordinates* $\mathbf{c} = (c_1, c_2, c_3, \ldots, c_K)$ *with* $\forall i \in 1..K, c_i \in 1..div$, *can be mapped to a unique* root region *of* $\mathbf{c}$ *with co-ordinates* $\mathbf{rt_c} = (c_1 - a_c, c_2 - a_c, c_3 - a_c, \ldots, c_K - a_c)$, *where* $a_c = \min(\{c_i \mid i \in 1$
Notice that a root region has the property that at least one of its co-ordinates has the value 1. *The concepts here are visualized in Figure 18.*

Lemma B.2 *Two regions with co-ordinates* $\mathbf{c} = (c_1, c_2, c_3, \ldots, c_K)$ *and* $\mathbf{d} = (d_1, d_2, d_3, \ldots, d_K)$, *with* $\forall i \in 1..K, c_i < d_i$, *share the same root region if and only if* $\exists b \in \mathbb{N}^+, \forall i \in 1..K, d_i = c_i + b$, *and* $\forall i \in 1..K (c_i, d_i \in 1..div)$

Proof B.2 *We have:*
$\mathbf{c} = (c_1, c_2, c_3, \ldots, c_K)$ *and* $\mathbf{d} = (c_1 + b, c_2 + b, c_3 + b, \ldots, c_K + b)$ *and* $a_c = \min(\{c_i \mid i \in 1..K\}) - 1, a_d = \min(\{c_i + b \mid i \in 1..K\}) - 1$.
Clearly, $a_c = a_d - b$.
Now, $\mathbf{rt}_c = (c_1 - a_c, c_2 - a_c, c_3 - a_c, \ldots, c_K - a_c)$ *and* $\mathbf{rt}_d = (c_1 + b - a_d, c_2 + b - a_d, c_3 + b - a_d, \ldots, c_K + b - a_d)$
By substituting $a_d - b$ *for* a_c *in the expression for* $\mathbf{rt}_c$ *we have:*
$\mathbf{rt}_c = (c_1 + b - a_d, c_2 + b - a_d, c_3 + b - a_d, \ldots, c_K + b - a_d) = \mathbf{rt}_d$, *proving the sufficient condition.*

To prove the necessary condition:
Assume $\mathbf{rt}_c = \mathbf{rt}_d$ *and* $\mathbf{c} = (c_1, c_2, c_3, \ldots, c_K)$ *and* $\mathbf{d} = (c_1 + b_1, c_2 + b_2, c_3 + b_3, \ldots, c_K + b_K)$ *and* $\neg(b_1 = b_2 = b_3 = \ldots = b_K)$. *Now* $\mathbf{rt}_c = (c_1 - a_c, c_2 - a_c, c_3 - a_c, \ldots, c_K - a_c)$ *and*
$\mathbf{rt}_d = (c_1 + b_1 - a_d, c_2 + b_2 - a_d, c_3 + b_3 - a_d, \ldots, c_K + b_K - a_d)$
Since $\mathbf{rt}_c = \mathbf{rt}_d$, $(b_1 - a_d, b_2 - a_d, b_3 - a_d, \ldots, b_K - a_d) = (-a_c, -a_c, -a_c, \ldots - a_c)$ *which implies that* $b1 = b2 = b3 = \ldots = b_K$, *a contradiction.*

Lemma B.3 *If two regions have different co-ordinates but map to the same root region then one is superior to the other.*

Proof B.3 *From lemma B.2 the two regions must have co-ordinates* $\mathbf{c} = (c_1, c_2, c_3, \ldots, c_K)$ *and* $\mathbf{d} = (c_1 + b, c_2 + b, c_3 + b, \ldots, c_K + b)$ *with* $b \in \mathbb{N}^+$. *Therefore c is superior to d because its co-ordinates are lower in every dimension.*

Corrollary B.1 *In any set of mutually non-inferior regions, each region must map to a different root region. This follows from lemma B.3.*

Theorem B.1 *The maximum number of mutually non-inferior regions in a K dimensional vector space divided up into div equal divisions in each dimension, is* $div^K - (div - 1)^K$.

Proof B.4 *We can see that there are exactly* $div^K - (div-1)^K$ *root regions in the vector space from lemma B.1 and the definition of a root region. Since every non-inferior region maps to a different root region B.2 then the maximum number of non-inferior regions is also* $div^K - (div-1)^K$.

C Derivation of Equation 7

Reproduced from [Rob, 2001]

Q. What is the number of ways of getting the sum s on n dice with x faces each?

A. The generating function for one die with x sides is

$$f(z) = z + z^2 + z^3 + \ldots + z^x. \tag{9}$$

The coefficient of a term tells you how many ways you can roll the exponent of z in the term. In this case, for numbers from 1 to x, you can roll each in one way. For other numbers, like 0 and numbers greater than x, you can roll them in zero ways. Thus the above generating function is right for one die. For two dice, square the above function. For n dice, raise it to the nth power. Then you want the coefficient of z^s in that expression. When calculating this coefficient for some specific value of s, you can do the arithmetic and ignore all powers of z with exponents bigger than s. If you want all the probabilities, you can ignore all exponents bigger than $(n.x+1)/2$, because the number of ways of rolling s is the same as the number of ways of rolling $n.x+1-s$, and if one is larger than $(n.x+1)/2$, then the other is smaller.

To express this in binomial coefficients, you can write:

$$\begin{aligned}
f(z) &= (z - z^{x+1})/(1-z) && (10)\\
f(z)^n &= [(z - z^{x+1})/(1-z)]^n && (11)\\
&= z^n.(1-z^x)^n.(1-z)^{-n} && (12)\\
&= z^n.\sum_{k=0}^{n}(-1)^k.\binom{n}{k}.z^{(x.k)}.\sum_{i=0}^{n}\binom{n+i-1}{i}.z^i && (13)\\
&= \sum_{k=0}^{n}\sum_{i=0}^{n}(-1)^k.\binom{n}{k}.\binom{n+i-1}{n-1}.z^{(n+x.k+i)} && (14)
\end{aligned}$$

Now the coefficient of z^s in this will be given by:

$$\sum_{k=0}^{(s-n)/x}(-1)^k.\binom{n}{k}.\binom{s-x.k-1}{n-1} \tag{15}$$

Algorithm: $AA_{\mathsf{Reduce}\in RED}$
M_t is the nondominated vectors archive
$\mathsf{Gen}(t)$ is a generating function with positive generation probability for all feasible vectors
$\mathbf{z}_t$ is the objective vector generated at time t

$t \leftarrow 0$
$M_t \leftarrow \emptyset$
while(1)
{
  $t \leftarrow t + 1$
  $\mathbf{z}_t \leftarrow \mathsf{Gen}(t)$ /* *Generate new vector* */
  $M_t \leftarrow \mathsf{Reduce}(\{\mathbf{z}_t\} \cup M_{t-1})$/* *Use the* Reduce *function to update* M_t */
}

Figure 1: Generic Archiving Algorithm $AA_{\mathsf{Reduce}\in RED}$, where RED is the set of all Reduce functions

Reduce Function: $\mathsf{Unbounded}(\{\mathbf{z}_t\} \cup M_{t-1}) \in RED$

M_{t-1} is the nondominated vectors archive after the last update
M_t is the new nondominated vectors archive
$\mathbf{z}_t$ is the objective vector generated at time t

$M_t \leftarrow \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$
return (M_t)

Figure 2: The Reduce function for storing an unbounded archive of all nondominated vectors.

Reduce Function: $\mathsf{Bounded}(\{\mathbf{z}_t\} \cup M_{t-1}) \in RED$

M_{t-1} is the nondominated vectors archive after the last update
M_t is the new nondominated vectors archive
$\mathbf{z}_t$ is the objective vector generated at time t
$arcsize$ is a bound on the archive's size

if $(\mathbf{z}_t < M_{t-1})$
  $M_t \leftarrow \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$
else if $(|M_{t-1}| < arcsize)$
  $M_t \leftarrow \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$
else
  $M_t \leftarrow M_{t-1}$
return (M_t)

Figure 3: The Reduce function for a simple bounded archive.

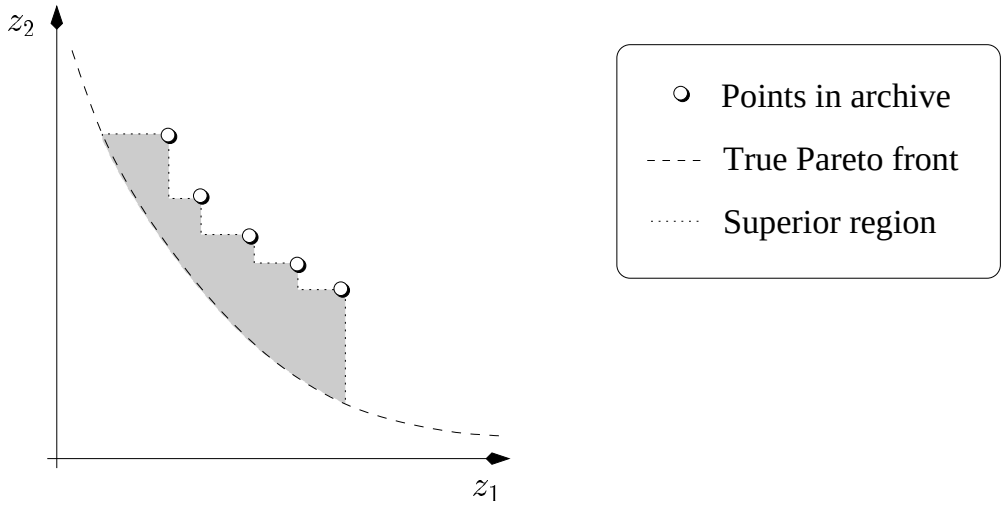

Figure 4: In an efficiency preserving strategy, only the superior region (shaded area) of a set of points is reachable. Thus, in this example only a restricted region of the Pareto front can be obtained.

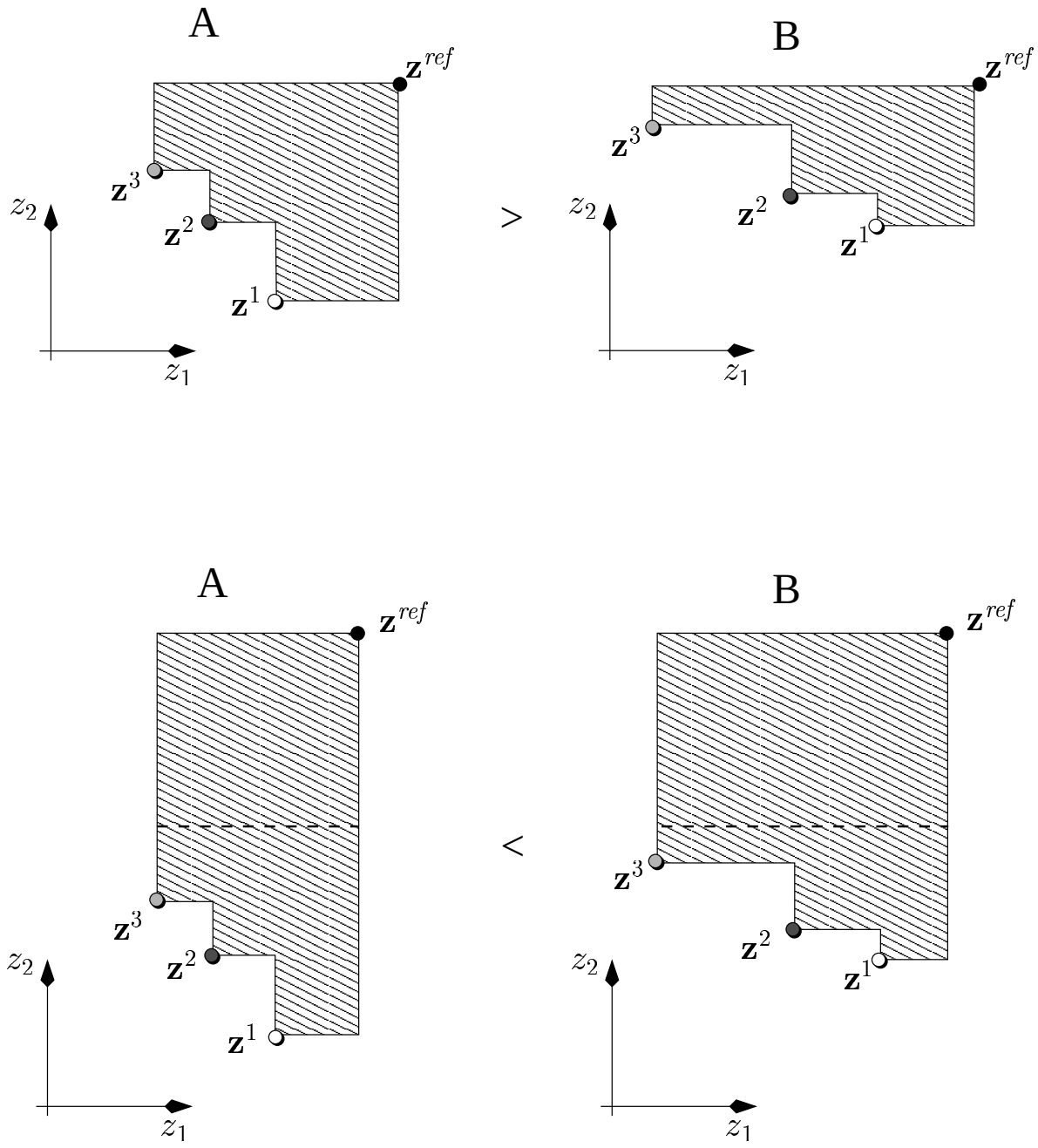

Figure 5: The relative value of the $\mathcal{S}$ metric depends upon a rather arbitrary choice of where the reference point is chosen. In the upper half of the figure two nondominated point sets are shown, A and B. With the chosen reference point $\mathcal{S}(A) > \mathcal{S}(B)$. But in the lower half the same point sets have a different ordering in $\mathcal{S}$. The reference point has been moved to a larger value in objective 2 and a smaller value in objective 1, and consequently $\mathcal{S}(A) < \mathcal{S}(B)$.

size_of_set$(A, \mathbf{z}^{ref}, k)$

$\mathbf{z}^{high}$ is the vector with the largest value in objective k from amongst the vectors in the updated set A
NDk(A, k) returns the nondominated vectors from the set A with respect to the lowest k objectives only.

$\mathcal{S} \leftarrow 0$
$z_k^{prev} \leftarrow z_k^{ref}$
while$(A \neq \emptyset)$
    $A \leftarrow$ NDk$(A, k-1)$
    if$(k < 3)$
        $\mathcal{S}_{k-1} \leftarrow z_1^{high}$
    else
        $\mathcal{S}_{k-1} \leftarrow$ size_of_set$(A, \mathbf{z}^{ref}, k-1)$
    $\mathcal{S} \leftarrow \mathcal{S} + \mathcal{S}_{k-1}.|z_1^{high} - z_1^{prev}|$
    $z_1^{prev} \leftarrow z_1^{high}$
    $A \leftarrow A \setminus \{\mathbf{z}^r \mid z_1^r \geq z_1^{high},\ \mathbf{z}^r \in A\}$
return $\mathcal{S}$

Figure 6: Recursively computing the $\mathcal{S}$ metric of a nondominated set A, in k objectives. The function has $O(n^{k+1})$ complexity in general, where n is the number of points in the set A.

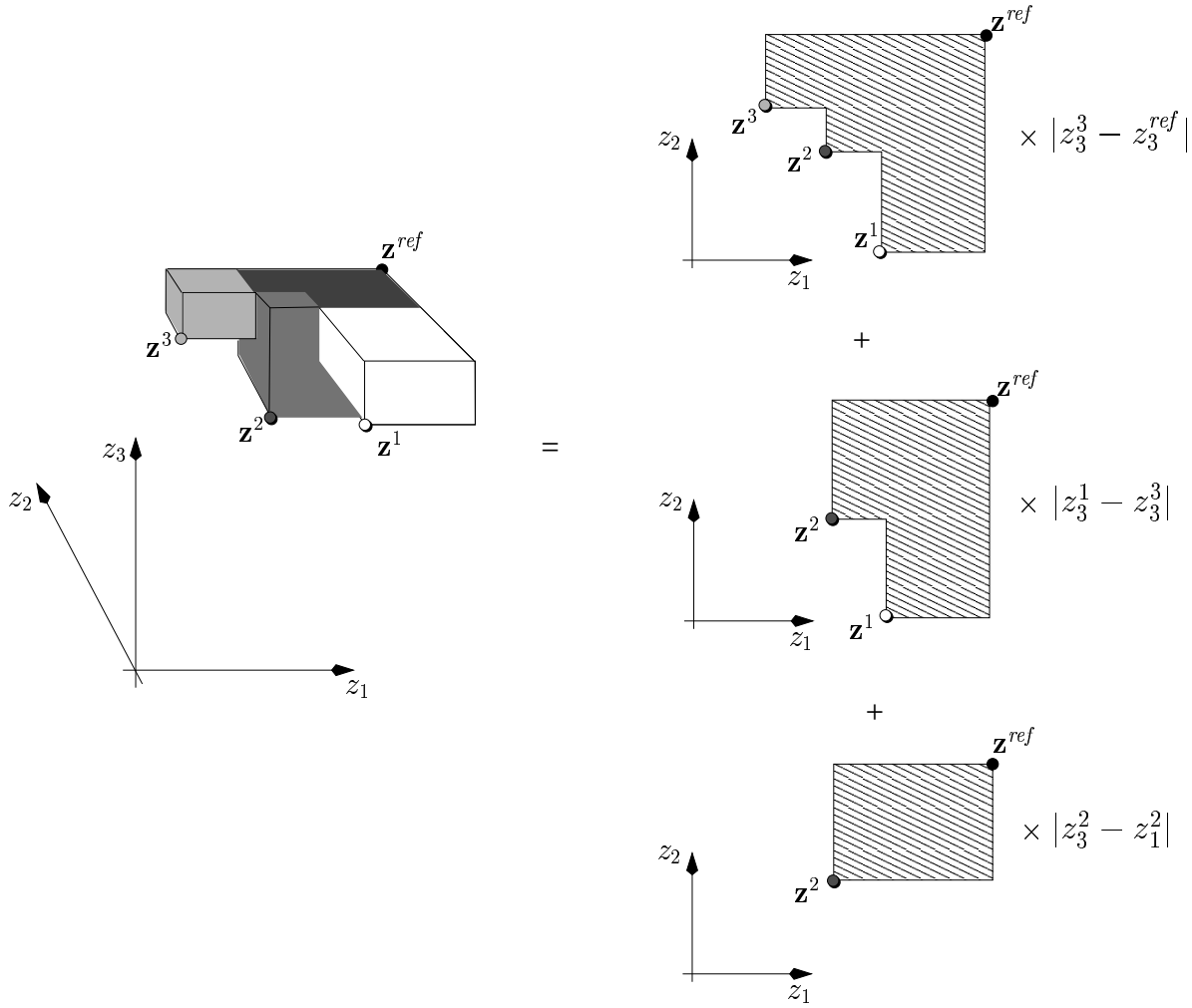

Figure 7: How the $\mathcal{S}$ metric is calculated in multiple dimensions. Three nondominated points in a 3-objective space are shown (left). The regions they weakly dominate, and which weakly dominate the reference point, are shown by the shaded cuboids. The combined size of these regions is shown by the calculation on the right.

Reduce Function: $\mathsf{S}(\{\mathbf{z}_t\} \cup M_{t-1}) \in RED$

M_{t-1} is the nondominated vectors archive after the last update
M_t is the new nondominated vectors archive
$\mathbf{z}_t$ is the objective vector generated at time t
$\mathcal{S}(Z)$ is the $\mathcal{S}$ metric value of a set of vectors, Z, assuming a suitable reference vector
$\mathbf{z}^{(min)}$ is randomly selected from $Z^{min} \subseteq M_{t-1}$ and
$Z^{min} = \{\mathbf{z}' \in M_{t-1} \mid \forall \mathbf{z} \in M_{t-1}, \mathcal{S}(M_{t-1} \cup \{\mathbf{z}_t\} \setminus \{\mathbf{z}'\}) \geq \mathcal{S}(M_{t-1} \cup \{\mathbf{z}_t\} \setminus \{\mathbf{z}\})$

---

if $(\mathbf{z}_t > M_{t-1})$ (Rule: Dominated)
  $M_t \leftarrow M_{t-1}$
else
  if $(\mathbf{z}_t < M_{t-1})$
    $M_t \leftarrow \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$ (Rule: Dominates)
  else if $(|M_{t-1}| = arcsize)$
    if $\mathcal{S}((M_{t-1} \cup \{\mathbf{z}_t\}) \setminus \{\mathbf{z}^{(min)}\}) > \mathcal{S}(M_{t-1})$
      $M_t \leftarrow (M_{t-1} \cup \{\mathbf{z}_t\}) \setminus \{\mathbf{z}^{(min)}\}$ (Rule: Size)
    else
      $M_t \leftarrow M_{t-1}$ (Rule: Steady_State)
  else
    $M_t \leftarrow \mathsf{ND}(M_{t-1} \cup \{\mathbf{z}_t\})$ (Rule: Fill)
return (M_t)

Figure 8: The Reduce function for a bounded archive that seeks to maximize the $\mathcal{S}$ metric value of the objective space.

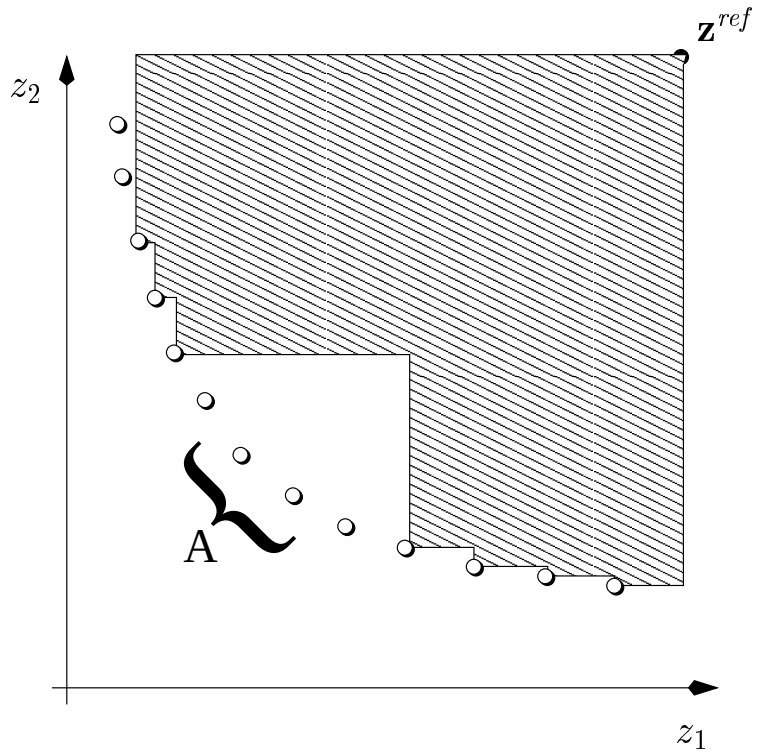

Figure 9: The figure shows the true Pareto front (white points) of a minimization problem. The hatched area shows the region dominated by seven of the Pareto optimal points. These points represent the (full) archive of an algorithm using $\mathcal{S}$ metric archiving. Clearly, these points in the archive are not as well-distributed as they could be. Fortunately, this seems to imply that they are not a local optimum of $\mathcal{S}$. A net increase in $\mathcal{S}$ would result if any vector pointed at by the label A were to replace any other vector in the current archive.

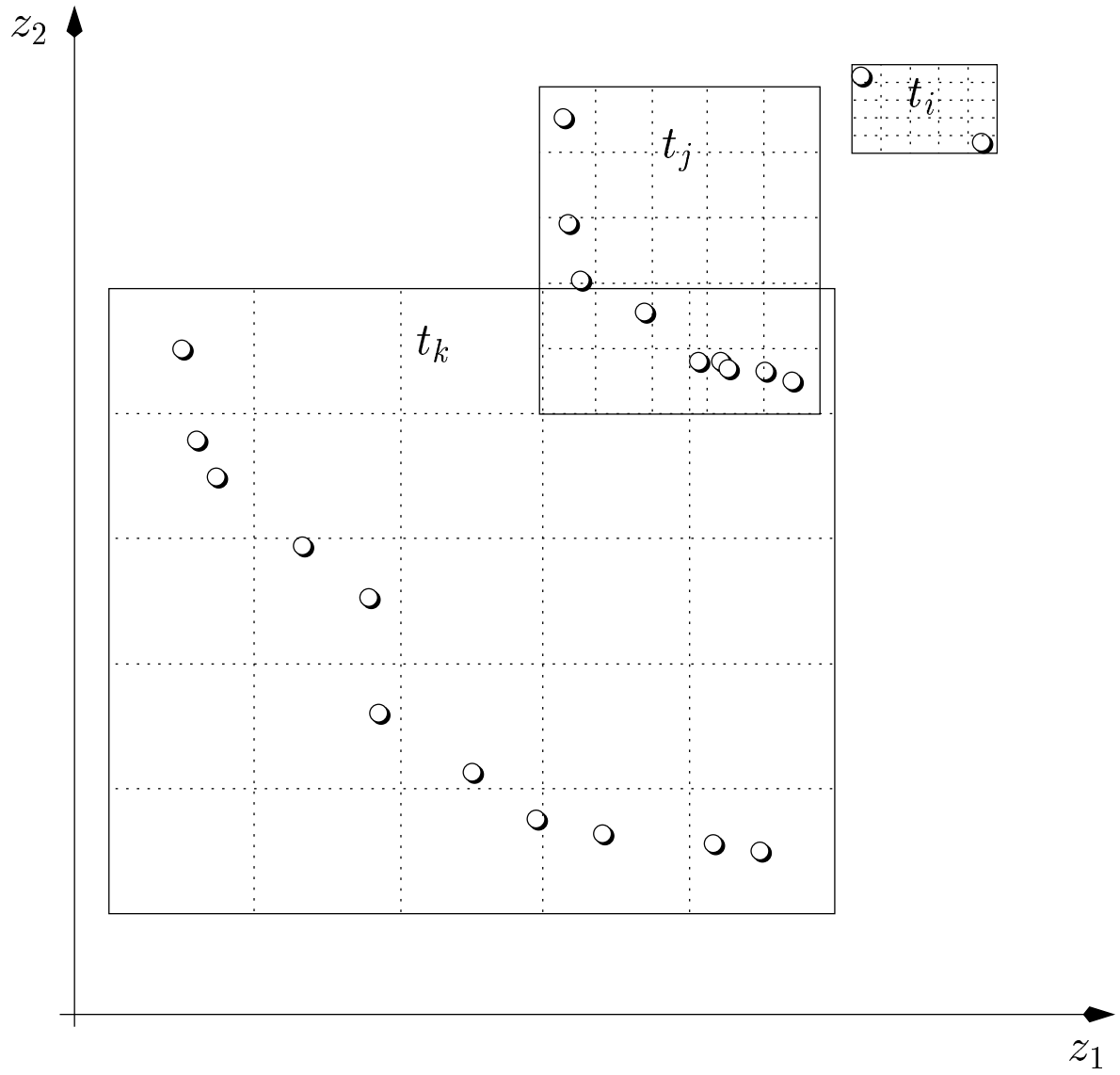

Figure 10: How the adaptive grid changes its location and shape in objective space as the vectors in the archive M_t change through iterations $t_i < t_j < t_k$.

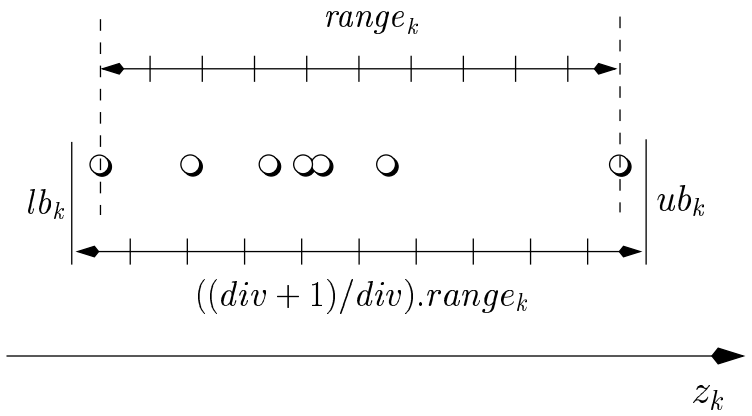

Figure 11: When a new vector $\mathbf{z}_t$ increases the range of the grid in some objective k, a new $range_{k,t}$ is calculated, and the grid boundaries are set so that $range_{k,t} \leftarrow \max_{z \in N_t}(z_k) - \min_{z \in N_t}(z_k)$ and $ub_{k,t} \leftarrow \max_{z \in N_t}(z_k) + (1/(2.div))(range_{k,t})$ and $lb_{k,t} \leftarrow \min_{z \in N_t}(z_k) - (1/(2.div))(range_{k,t})$, which means that the two extremal vectors in objective k will be located at the center of the outer grid regions.

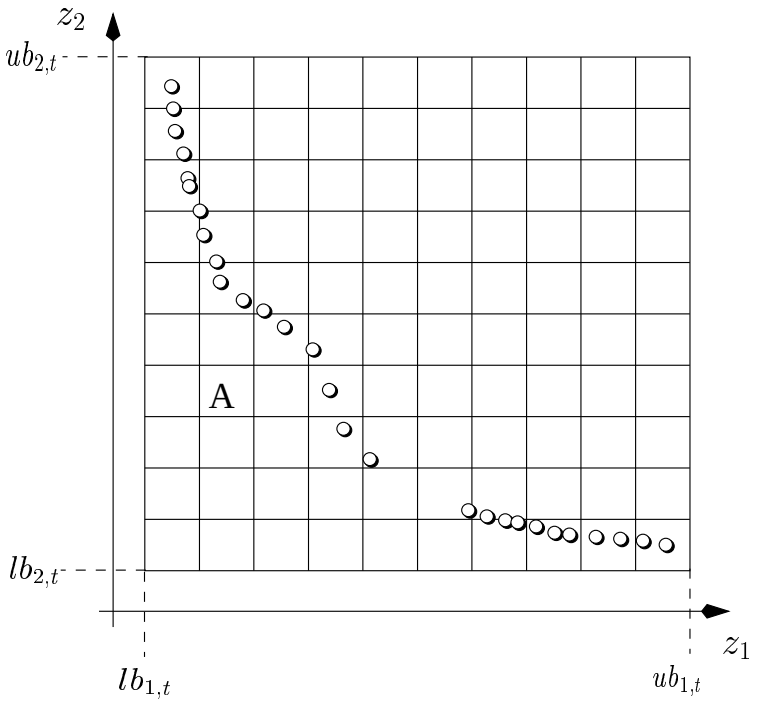

Figure 12: The figure shows the meaning of $ub_{k,t}$ and $lb_{k,t}$, for a two objective plot. The region labelled A has co-ordinates $\mathbf{i} = (2, 4)$ and boundaries, $rub_{1,\mathbf{i},t} = (2/10).(ub_{1,t} - lb_{1,t})$, $rlb_{1,\mathbf{i},t} = (1/10).(ub_{1,t} - lb_{1,t})$, $rub_{2,\mathbf{i},t} = (4/10).(ub_{2,t} - lb_{2,t})$ and $rlb_{2,\mathbf{i},t} = (3/10).(ub_{2,t} - lb_{2,t})$.

Reduce Function: Adaptive_Grid_Archiving $\in RED$

M_t is the nondominated vectors archive
Gen(t) is a generating function with positive generation probability for all feasible vectors
$\mathbf{z}_t$ is the objective vector generated at time t
$\mathbf{z}^{c,t}$ is a vector in M_{t-1}, selected from the set of most crowded regions, at random (see Definition 4.15)

---

foreach $(k \in 1..K)$ (Rule: Update_Boundaries)
    $range_{k,t} \leftarrow \max_{z \in N_t}(z_k) - \min_{z \in N_t}(z_k)$
    $ub_{k,t} \leftarrow \max_{z \in N_t}(z_k) + (1/(2.div))(range_{k,t})$
    $lb_{k,t} \leftarrow \min_{z \in N_t}(z_k) - (1/(2.div))(range_{k,t})$
    Re-calculate all grid region boundaries for dimension k
if $(\mathbf{z}_t > M_{t-1})$
    $M_t \leftarrow M_{t-1}$ (Rule: Dominated)
else
    if $(\mathbf{z}_t < M_{t-1})$
        $M_t \leftarrow M_{t-1} \cup \{\mathbf{z}_t\} \setminus \{\mathbf{z} \in M_{t-1} \mid \mathbf{z}_t < \mathbf{z}\}$ (Rule: Dominates)
    else if $(|M_{t-1}| = arcsize)$
        if ($\mathbf{z}_t$ outside old boundaries of grid)
            $M_t \leftarrow M_{t-1} \cup \{\mathbf{z}_t\} \setminus \{\mathbf{z}^{c,t}\}$ (Rule: Extends)
        else if ($\mathbf{z}_t$'s region is not a most crowded region $\wedge\ \exists \mathbf{i}, p_{nue}(r^{\mathbf{i}}_{t-1} > 1)$
            $M_t \leftarrow M_{t-1} \cup \{\mathbf{z}_t\} \setminus \{\mathbf{z}^{c,t}\}$ (Rule: Reduce_Crowding)
        else
            $M_t \leftarrow M_{t-1}$ (Rule: Steady_State)
    else
        $M_t \leftarrow M_{t-1} \cup \mathbf{z}_t$ (Rule: Fill)
return(M_t)

Figure 13: The Reduce function for Adaptive Grid Archiving

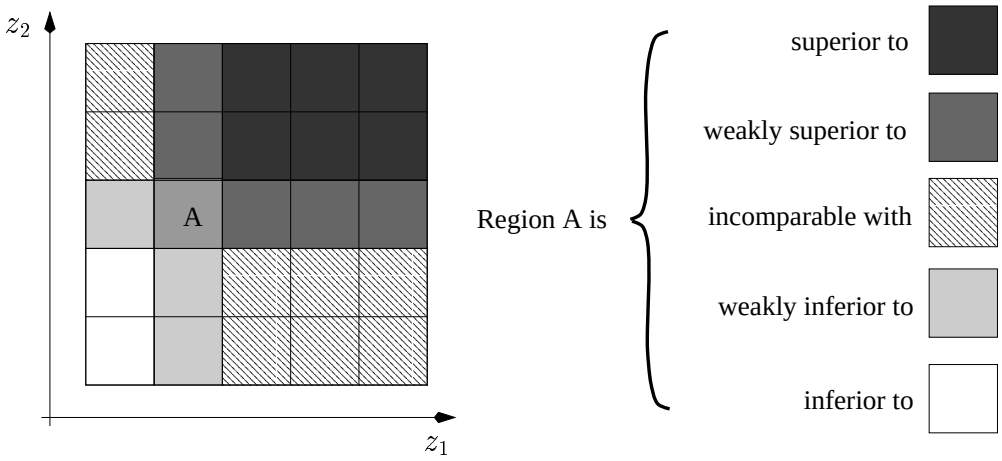

Figure 14: The dominance relationships between a region A and the other regions in a grid

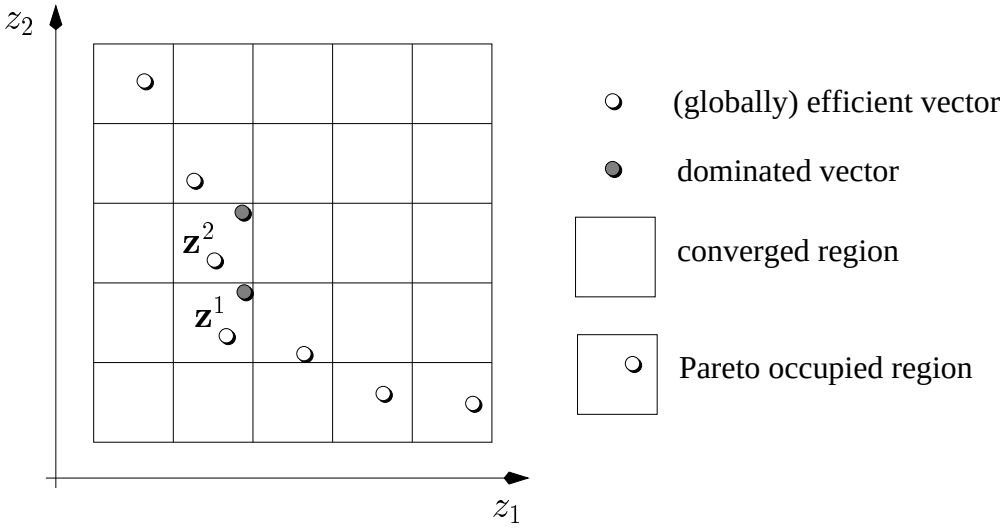

Figure 15: A set of (globally) efficient vectors (white points) within a space of converged regions is shown. Although both vectors $\mathbf{z}^1$ and $\mathbf{z}^2$ are nondominated, the region that $\mathbf{z}^1$ occupies, $r^{2,2}$, is weakly superior to the region $\mathbf{z}^2$ occupies, $r^{2,3}$. Because $r^{2,3}$ is weakly inferior to $r^{2,2}$, there are points in $r^{2,3}$ that are dominated by points in $r^{2,2}$, e.g., the grey point in $r^{2,2}$ is dominated by $\mathbf{z}^1$. Region $r^{2,2}$, in contrast, is a critical Pareto occupied region because it is not weakly inferior to any other Pareto occupied region. This means that any vector in it cannot be dominated by any feasible vector in any other region. Consider the grey vector in $r^{2,2}$. Although it is dominated (by $\mathbf{z}^1$), clearly it cannot be dominated by any vector not in $r^{2,2}$

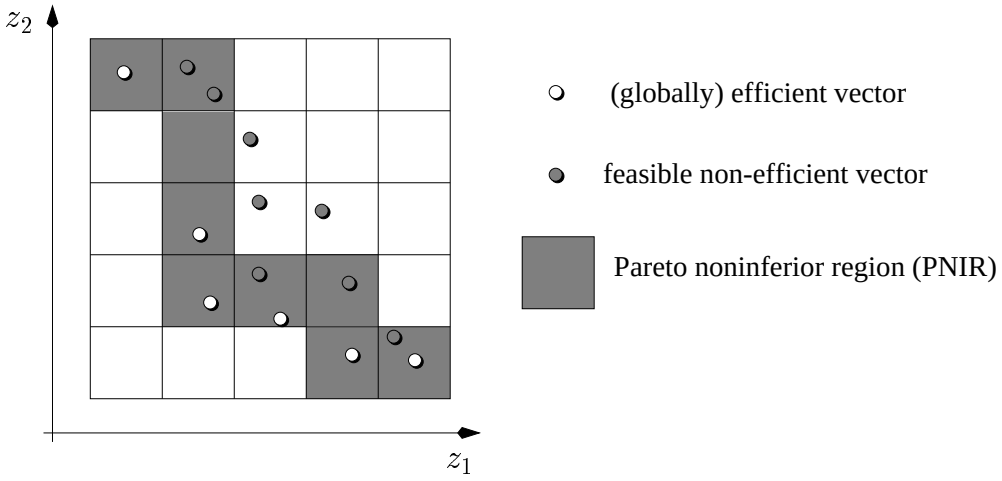

Figure 16: The feasible objective vectors within a converged set of regions is shown. The Pareto non-inferior regions ($PNIR$s) are shaded.

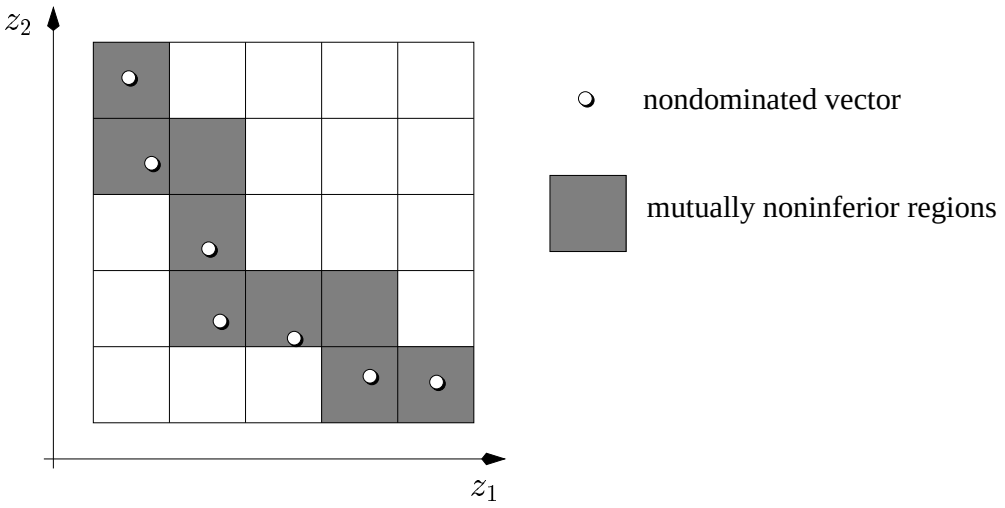

Figure 17: A set of mutually nondominated vectors is shown. All of them must lie within a set of regions that is mutually non-inferior.

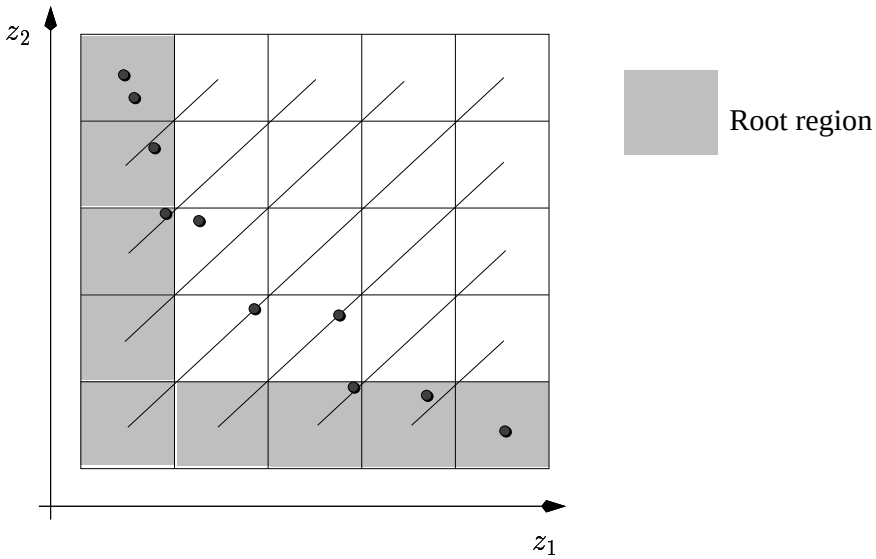

Figure 18: An illustration of root regions. Diagonal lines have been drawn to show all of the regions that map to the same root region. Clearly, if two vectors are in different regions on the same diagonal then one must be superior to the other. Therefore, any nondominated set (like the one shown) cannot occupy more regions than there are root regions

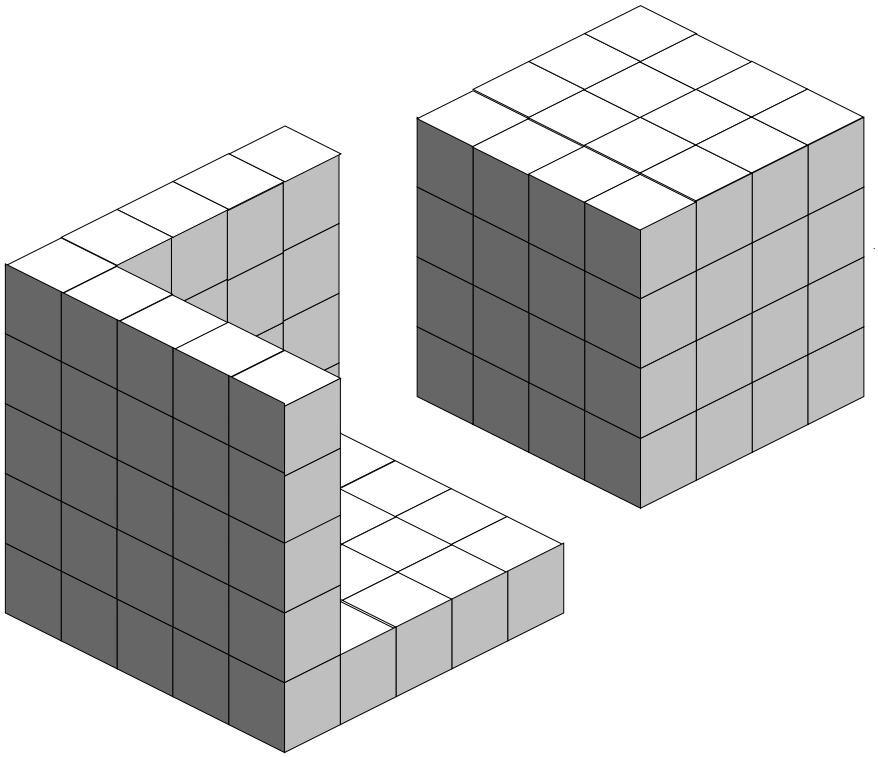

Figure 19: An illustration of the equation $div^K - (div - 1)^K$

K	div	$div^K - (div-1)^K + 2K$	$v_{\lfloor K/2(div+1) \rfloor}$
2	2	7	2
2	4	11	4
2	8	29	8
2	16	35	16
2	32	67	32
2	64	131	64
3	2	13	3
3	4	43	12
3	8	175	48
3	16	727	192
4	2	23	6
4	4	183	44
4	8	1703	344
4	16	14919	2736
8	2	272	70
8	4	58991	8092
8	8	11012431	1012664

Table 1: The required size of the archive for various values of div and K. The third column is the required size if convergence to all the critical Pareto occupied regions must be guaranteed. The fourth column is the required size to allow storage of vectors occupying the maximum number of critical Pareto occupied regions, but without guaranteeing that convergence to this set of regions will occur.