Internal Report 08-12 augustus 2008

Universiteit Leiden

Opleiding Informatica

“Multiobjective Optimization of Water Distribution Networks Using SMS-EMOA”

Edgar Reehuis

Bachelorscriptie

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Multiobjective Optimization of Water Distribution Networks Using SMS-EMOA

Edgar Reehuis
ereehuis@liacs.nl
Student ID: 0111996

September 9, 2008

Abstract

The multiobjective evolutionary algorithm SMS-EMOA was shown by Emmerich et al. to outperform the well-established NSGA-II on a range of common test problems, using the $\mathcal{S}$ metric as comparison criterion. This study assesses which of the two algorithms performs best with respect to the optimization of water distribution networks, using the unconstrained three objective reformulation by Formiga et al. of this originally constrained single objective problem. The well-known test problems Two Loop and Hanoi have been examined, showing the best results by SMS-EMOA in both cases.

1 Introduction

Optimization of the design of *water distribution networks* has been the subject of study by scientists in different fields for more than thirty years. The water distribution backbone of a city is modeled using *pipes*, *nodes*, *reservoirs*, *pumps*, and *valves*. In this model the pipes take up a central position as it is the selection of *diameter sizes* for pipe segments that governs the optimization process. Smaller diameters are less expensive to procure, while larger diameters result in higher water pressure at the demand nodes in the network.

In the original problem formulation, published in 1977 by Alperovits and Shamir [13], the design optimization was formulated as a *single objective* problem with *investment cost* as *objective* and a *constraint* on the mimimal *nodal pressure*; in 2003 Formiga et al. [1] expressed it as an *unconstrained multiobjective* problem, using three objectives: *Investment Cost*, *System Entropy* and *Demand Supply Ratio*. Investment Cost was inherited from the original formulation, while Demand Supply Ratio was included as an objective to maximize the mean nodal pressure. The third objective, System Entropy, was included to increase the reliability of the network by appreciating redundancy in the system; many existing optimization schemes were likely to assign small or null diameters to pipe segments, effectively changing the design from a *looped* network to a *branching* network [1].

Furthermore, a multiobjective approach has the potential of providing interesting solutions that are ignored when only Investment Cost is taken into account: imagine two solutions, one being slightly more expensive than the other but scoring better on both reliability and mean nodal pressure.

For their optimization Formiga et al. [1] used the well established *NSGA-II multiobjective evolutionary algorithm.* In 2005 Emmerich et al. [5] presented their *SMS-EMOA* multiobjective evolutionary algorithm. They showed it outperformed NSGA-II and a few other state-of-the-art optimizers on a range of two objective [5] and three objective [7] test problems. Now using the formulation by Formiga et al. [1] for multiobjective optimization of water distribution networks, it is to be determined how SMS-EMOA performs in relation to NSGA-II with respect to this real-world problem.

This work is structured as follows: in Section 2 the optimization problem is defined, zooming in on the different elements that comprise the optimization loop. Section 3 provides some definitions needed for multiobjective optimization, lays out the general multiobjective evolutionary algorithm, and gives insight into the differences between NSGA-II and SMS-EMOA. Next Section 4 deals with the problem specific adaptation needed for application of NSGA-II and SMS-EMOA to optimization of water distribution networks. Section 5 gives an overview of the test problems that were examined and then in Section 6 the results of this examination are analyzed. Finally Section 7 provides concluding remarks and some input for future work.

2 Problem Definition

Starting from a test problem with a fixed *topology* (i.e. the node and pipe layout) and *topography* (i.e. the *elevations* of the nodes and reservoir), the optimization problem is to select the best diameter sizes for the pipes in the network from a table of *available commercial diameters.* From a reliability and nodal pressure perspective, the largest diameter available is ideally used throughout the network as this will cause the least *friction loss* in the pipes and provide the maximum number of good alternative routes; however, this will also be the most expensive solution available. Thus multiobjective optimization is used, leading to a set of mutually *non-dominated* solutions, including *good compromise* solutions based on the preferences the designer has for one or more objectives.

The optimization loop has three main components (see Figure 1):

1. *Hydraulic simulator* (see Section 2.1): an optimizer which solves the non-linear equation system by minimizing residuals (see Section 2.1.4), taking as input the network definition (i.e. topology and topography) and one of the solutions generated by the *multiobjective optimizer* at a

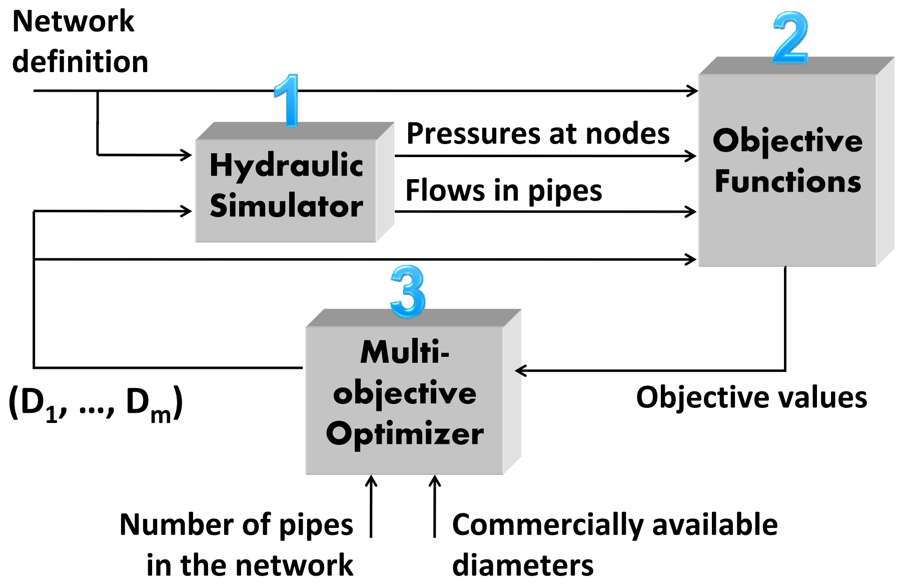


Figure 1: The Multiobjective Optimization Loop

time (indicating the selected diameter size per pipe), while providing *flows* and *pressures* as output (to be used as input by the *objective functions*);

2. *Objective functions* (see Section 2.2): calculate the fitness of a solution, expressed in three objectives, based on the network definition, the flows and pressures received from the hydraulic simulator (depending on the current diameter selection), and the current diameter selection determined by the solution that is being assessed;

3. *Multiobjective optimizer* (see Section 3): generates new solutions, which are vectors of diameter codes (the codes point to certain size and price couples in the table of commercial diameters), and selects the best solutions to proceed to the next *generation* based on their *objective values*.

2.1 Hydraulic Simulator

The hydraulic simulator models the *water distribution network* consisting of *nodes*, *reservoirs*, and *pipes*. It produces two kinds of output: the *hydraulic heads* (i.e. potential energy of the water due to elevation) at the nodes and the *flows* in the pipes. In this study the *EPANET 2* simulator by Rossman [2] was used. Starting from a set of known hydraulic heads at the *reservoir* nodes (namely: the *elevation* of the reservoir), it calculates the hydraulic head at each *internal* node and the flow in each pipe (while compensating for *friction loss* depending on the choice of *diameters* for the pipes).

2.1.1 Constants

The following constants are provided in the definition of a test problem:

m	number of *pipes* in the network;
r	number of *reservoir* nodes in the network;
n	number of *internal* nodes in the network;
$r+n$	total number of nodes in the network;
L_i	length of pipe i;
b_i	begin node of pipe i;
e_i	end node of pipe i;
C_i	roughness coefficient of pipe i, for use in the *friction head loss formula* (see Section 2.1.5);
Z_j	elevation of node $j \in \{1, \ldots, r+n\}$;
Q_j	water demand drawn from internal node $j \in \{r+1, \ldots, r+n\}$;
p^{zero}	pressure corresponding to zero nodal water demand satisfaction;
p^{req}	pressure required to satisfy the nodal water demand completely;
N^{diam}	number of diameters in the table of available commercial diameters.

For an illustration of a test problem network, see Figure 2.

2.1.2 Simulation Variables

The following variables are calculated or can change during a run of the hydraulic simulator:

p_j effective pressure remaining at internal node $j \in \{r+1, \ldots, r+n\}$;

h_j hydraulic head (i.e. potential energy of the water) at node $j \in \{1, \ldots, r+n\}$, where:

$$h_j = \begin{cases} Z_j & j \leq r \\ Z_j + p_j & else \end{cases} \tag{1}$$

q_i water flow in pipe i (from $orig(i)$ to $dest(i)$);

$orig(i)$ origin of the water flow in pipe i, where:

$$orig(i) = \begin{cases} b_i & q_i \geq 0 \\ e_i & q_i < 0 \end{cases} \tag{2}$$

$dest(i)$ destination of the water flow in pipe i, where:

$$dest(i) = \begin{cases} e_i & q_i \geq 0 \\ b_i & q_i < 0 \end{cases} \tag{3}$$

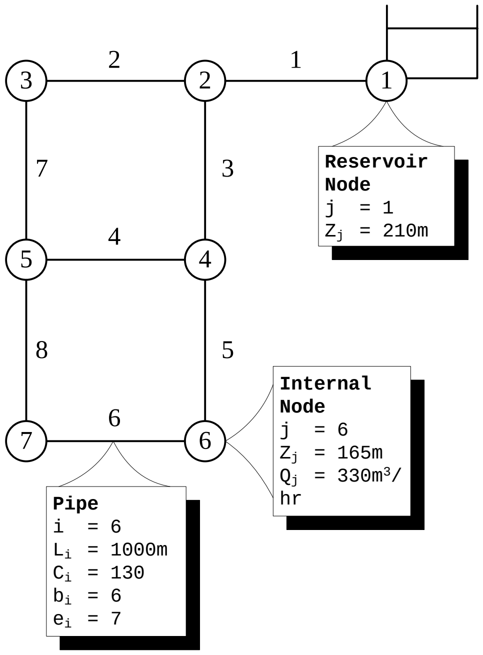


Figure 2: The Two Loop Network: node data, pipe data, and available commercial diameters are listed in Table 6, 7, and 8 in the Appendix.

2.1.3 Decision Variables

The following variables are varied during the optimization process:

D_i diameter code of pipe i, indicating the selected size/price entry from the table of available commercial diameters (supplied with the test problem, for an example see Table 8 in the Appendix).

2.1.4 Equations

Two sets of equations have to be satisfied:

- *Hydraulic head loss* (i.e. pressure loss) in all pipes i:

$$loss_i = h_{orig(i)} - h_{dest(i)} = K_i |q_i|^\alpha \tag{4}$$

where:

K_i resistance coefficient of pipe i, depending on pipe roughness, length, and the selected diameter (i.e. $K_i \equiv K(C_i, L_i, D_i)$); the exact form of this coefficient is determined by the used *friction head loss formula* (see Section 2.1.5);

α exponent of the friction head loss formula.

- *Flow continuity* around all internal nodes $j \in \{r+1, \ldots, r+n\}$:

$$f_j = \sum_{i:\, j=dest(i)} |q_i| - \sum_{i:\, j=orig(i)} |q_i| - Q_j = 0 \tag{5}$$

  This rule basically means that the flow into a node must equal the flow going out of that node.

For a set of known hydraulic heads at the reservoir nodes, the remaining heads and flows are sought that satisfy equation 4 and 5. This can be done by combining these equations and reformulating the system as a minimization problem. First, isolate $|q_i|$ in equation 4:

$$|q_i| = \left(\frac{h_{orig(i)} - h_{dest(i)}}{K_i} \right)^{\frac{1}{\alpha}} \tag{6}$$

Then, replace all occurrences of $|q_i|$ in equation 5 by the righthand term of equation 6:

$$\begin{aligned} f_j = \sum_{i:\, j=dest(i)} \left(\frac{h_{orig(i)} - h_{dest(i)}}{K_i} \right)^{\frac{1}{\alpha}} - \\ \sum_{i:\, j=orig(i)} \left(\frac{h_{orig(i)} - h_{dest(i)}}{K_i} \right)^{\frac{1}{\alpha}} - Q_j = 0 \end{aligned} \tag{7}$$

Next we define the following system calculating the sum over all n squared errors around zero (i.e. solves equation 7 for every internal node):

$$\sum_{j=r+1}^{r+n} (f_j - 0)^2 \tag{8}$$

Now we can express the hydraulic simulator as a minimization problem solving the non-linear equation system by minimizing residuals; a tuple of hydraulic heads at the internal nodes $j \in \{r+1, \ldots, r+n\}$ is sought that minimizes the sum of the n squared errors around zero:

$$\underset{(h_{r+1},\ldots,h_{r+n}) \in \mathbb{R}^n}{\arg\min} \sum_{j=r+1}^{r+n} (f_j(h_1, \ldots, h_{r+n}))^2 \tag{9}$$

2.1.5 Friction Head Loss Formula

In an ideal setting, the hydraulic head remains the same at all nodes and does not decrease during transportation of the water through the network. In a real situation however, some potential energy of the water is converted into heat due to friction in the pipes. The friction head loss formula calculates this pressure loss. In this study the widely adopted, empirically derived *Hazen-Williams equation* [3] is used:

$$loss_i = \omega \frac{L_i}{{C_i}^{\alpha} size(D_i)^{\beta}} |q_i|^{\alpha} \tag{10}$$

where:

ω	numerical conversion constant, depending on the units used (see Section 2.1.6);
α	exponent taken to be 1.852 by Rossman [2];
β	exponent taken to be 4.871 by Rossman [2];
$size(D_i)$	diameter size corresponding to diameter code D_i.

The resistance coefficient K_i of pipe i (mentioned in Section 2.1.4) then becomes:

$$K_i = \omega \frac{L_i}{{C_i}^{\alpha} size(D_i)^{\beta}} \tag{11}$$

2.1.6 Units

In principle arbitrary units may be used in a test problem definition (see Section 5) for each of its properties. As a result, different sets of units are used in different test problems. Dependences exist between certain properties however, with the requirement of using the same unit for both. If a dependency exists but different units are used, values will have to be converted using conversion constants[1].

The following dependences exist and they determine which units are used in the actual calculation (see Section 2.1.1, 2.1.2, and 2.1.3 for used notation):

- $Z_j \Rightarrow \begin{cases} h_j \\ L_i \\ p^{zero}, p^{req}, p_j \end{cases}$
- $q_i \Rightarrow Q_j$
- $size(D_i) \wedge q_i \Rightarrow \omega$

Per property we discriminate between the unit used in the calculation (*calculation* unit) and the unit defined in the test problem (*defined* unit). Inputs for which dependences exist (e.g. L_i, Q_j, ω) will have to be converted from the defined unit to the calculation unit prior to calculation; outputs for which dependences exist (e.g. p_j) will have to be converted from the calculation unit to the defined unit after calculation.

[1] For an exhaustive listing, visit `http://www.asknumbers.com`

The numerical conversion constant ω is a special input case as it is unitless. Rossman [2] defines ω as 4.727 for $size(D_i)$ in *feet* and q_i in *cubic foot per second*. The numerical constant can be converted for usage with different units as follows:

$$\omega = \frac{convQ^{\alpha}}{convD^{\beta}} \cdot 4.727 \tag{12}$$

where:

$convQ$	conversion constant from defined unit for q_i to *feet*;
$convD$	conversion constant from defined unit for $size(D_i)$ to *cubic foot per second*.

Both test problems that was applied optimization to (see Section 5) define $size(D_i)$ in *inch* and q_i in *cubic meter per hour*. This combination is not available in EPANET, so the available combination of $size(D_i)$ in *millimeters* and q_i in *cubic meter per hour* had to be used. The $size(D_i)$ values were converted prior to calculation. One *inch* exactly equals 25.4 *millimeter*, but we follow Formiga et al. [1] who use a conversion constant from *inch* to *millimeter* of 25, in order to be consistent in the test problem comparisons presented later in this work (see Section 6).

2.2 Objective Functions

Formiga et al. [1] define three objectives: *Investment Cost*, *System Entropy* and *Demand Supply Ratio*. Investment Cost depends on the (commercially available) diameter size selected per pipe and is, naturally, to be minimized.

System Entropy expresses the reliability of the network. The *water demand* at a given internal node is ideally met using multiple different paths to that node. The required *flow* should be distributed over these routes as evenly as possible. This way, should a segment of the network fail, alternative routes exist which could still supply a reasonable part of the demand. System Entropy is determined by the flows in the different pipes in the network, thus, like Investment Cost, it depends on the selection of diameters. The value for System Entropy is to be maximized.

The third objective, Demand Supply Ratio, tells us to what extent the water demand at the various internal nodes in the network is met. It depends on the *pressure* (i.e. *hydraulic head* minus *node elevation*) remaining when the water reaches the node. Pressure at an internal node is gained by the relative elevation of the reservoir node (compared to the elevation of the internal node); pressure at an internal node is lost due to friction in the pipes during transportation. Since the other values are given as constants (the node elevations are defined in the test problem), Demand Supply Ratio varies solely with the friction loss and is thereby again determined by the choice of diameter sizes. The Demand Supply Ratio is to be maximized.

2.2.1 Investment Cost

The first objective function is [1]:

$$F_1 = \min[\sum_{i=1}^{m} price(D_i)L_i] \tag{13}$$

where:

$price(D_i)$ unit price of the ith pipe; the prices of the diameter sizes are included in the table of available commercial diameters, supplied with the test problem.

2.2.2 System Entropy

Let in_j be the water flow transported towards node j by the incoming pipes i connected to it:

$$in_j = \sum_{i:\, j=dest(i)} |q_i| \tag{14}$$

Let IN be the sum of all incoming flows in the network:

$$\mathrm{IN} = \sum_{j=1}^{r+n} in_j \tag{15}$$

Let s_j be the entropy of node j:

$$s_j = \sum_{i:\, j=dest(i)} \frac{-|q_i|}{in_j} \ln \frac{|q_i|}{in_j} \tag{16}$$

Let S be the entropy of the entire network:

$$S = \sum_{j=1}^{n} \frac{in_j}{\mathrm{IN}} (s_j - \ln \frac{in_j}{\mathrm{IN}}) \tag{17}$$

For the second objective function, the exponential of S is taken (increasing the range, for easier comparison of different solutions) [1]:

$$F_2 = \max[\exp S] \tag{18}$$

2.2.3 Demand Supply Ratio

In order to determine the portion of its water demand a node is capable of servicing, a functional relationship between pressure and water demand at a node needs to be established. Let p_j be the effective pressure remaining at node j:

$$p_j = h_j - Z_j \tag{19}$$

Let ${Q_j}^{met}$ be the portion of the water demand met at node j:

$$ {Q_j}^{met} = \begin{cases} 0 & p_j < p^{zero} \\ Q_j \sqrt{\frac{p_j - p^{zero}}{p^{req} - p^{zero}}} & p^{zero} \leq p_j \leq p^{req} \\ Q_j & p_j > p^{req} \end{cases} \tag{20} $$

The third objective function then becomes the maximization of the average demand supply ratio [1]:

$$ F_3 = \max[\frac{1}{n} \sum_{j=r+1}^{r+n} \frac{{Q_j}^{met}}{Q_j}] \tag{21} $$

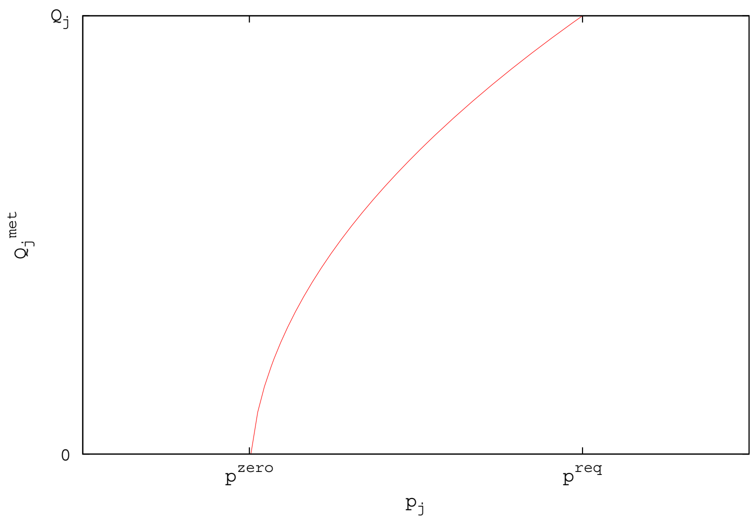


Figure 3: Plot of the functional relationship ${Q_j}^{met}$ for test problem Two Loop (i.e. $p^{zero} = 10, p^{req} = 30$; see Section 5.1).

3 Multiobjective Evolutionary Algorithms

Having explained the hydraulic simulator and defined the objective functions, let us now turn to the third component in the optimization loop: the multiobjective optimizer (see Figure 1). Formiga et al. [1] used the multiobjective evolutionary algorithm *NSGA-II* by Deb et al. [4] in their study. Emmerich et al. [5] showed that their *SMS-EMOA* outperforms NSGA-II for standard two dimensional benchmark problems ZDT1 through ZDT4 and ZDT6, concerning the $\mathcal{S}$ *metric* [8]. Moreover, Naujoks et al. [7] presented better $\mathcal{S}$ metric results for SMS-EMOA on three dimensional DTLZ problems. $\mathcal{S}$ metric or *hypervolume measure* is a commonly used comparison criterion for *non-dominated sets* or *Pareto fronts*. It expresses the volume in the multi-dimensional *objective space* that the points in the Pareto front jointly dominate, with respect to a *reference point*. For the problem of multi-objective optimization of water distribution networks, it is to be determined which of both algorithms performs best with respect to the $\mathcal{S}$ metric. In order to allow for a fair comparison, both NSGA-II and SMS-EMOA are implemented, enabling pairwise comparison of results for which the same same set of parameters was used (e.g. *population size*, *recombination probability*, *mutation probability*; see Section 4.3).

3.1 Definitions

A multiobjective optimization problem deals with solutions that exist in the decision space (i.e. they consist of decision variables) and that are ranked using their position in the objective space (i.e. their score for each of the objectives functions). The solution is sought that has the best values for each of the objectives (i.e. maximal value in case of a maximization objective; minimal value in case of a minimization objective). For determining the best of two solutions, the associated objective values will have to be compared pairwise. A solution can only be declared better than the other if at least one of its objective values is better and the rest of the objective values is not worse than those of its counterpart. In practice this means that not one best solution is found but a set of solutions for which no clear winner can be appointed. In the following paragraph these notions are formalized.

Zitzler and Thiele [8] define a multiobjective optimization problem as a vector function $\boldsymbol{f}$ that maps a tuple of v decision variables to a tuple of w objectives:

$$\arg\{\min_{\mathbf{x}}, \max\}\,\mathbf{y} = \boldsymbol{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_w(\mathbf{x})) \tag{22}$$

with:

$$\mathbf{x} = (x_1, \ldots, x_v) \in \mathbb{X} \tag{23}$$

$$\mathbf{y} = (y_1, \ldots, y_w) \in \mathbb{Y} \tag{24}$$

where:

$\mathbf{x}$ decision vector;
$\mathbb{X}$ decision space;
$\mathbf{y}$ objective vector;
$\mathbb{Y}$ objective space.

Assume a pure minimization problem (i.e. only minimization objectives), without loss of generality, and consider two decision vectors $\mathbf{x}^1, \mathbf{x}^2 \in \mathbb{X}$. We define the following order on the decision space: $\mathbf{x}^1$ is said to *dominate* $\mathbf{x}^2$ (in symbols: $\mathbf{x}^1 \prec \mathbf{x}^2$) iff:

$$\begin{aligned} &\forall i \in \{1,\ldots,w\} : f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2) \ \wedge \\ &\exists j \in \{1,\ldots,w\} : f_j(\mathbf{x}^1) < f_j(\mathbf{x}^2) \end{aligned} \tag{25}$$

Under the same assumptions, consider two objective vectors $\mathbf{y}^1, \mathbf{y}^2 \in \mathbb{Y}$. We then define a similar order on the objective space: $\mathbf{y}^1$ is said to *dominate* $\mathbf{y}^2$ (in symbols: $\mathbf{y}^1 \prec \mathbf{y}^2$) iff:

$$\begin{aligned} &\forall i \in \{1,\ldots,w\} : {y_i}^1 \leq {y_i}^2 \ \wedge \\ &\exists j \in \{1,\ldots,w\} : {y_j}^1 < {y_j}^2 \end{aligned} \tag{26}$$

The *efficient set* M consists of all decision vectors for which the corresponding objective vectors cannot be improved in any dimension without degradation in another [8]; all decision vectors for which no other decision vector exist that dominates it are contained [5]:

$$M = \{\mathbf{x}^1 \in \mathbb{X} | \nexists \mathbf{x}^2 \in \mathbb{X} : \mathbf{x}^2 \prec \mathbf{x}^1\} \tag{27}$$

The image $\mathcal{Y}_N$ of M in the objective space $\mathbb{Y}$ is called the *non-dominated set* or *Pareto front*:

$$\mathcal{Y}_N = \{\boldsymbol{f}(\mathbf{x}) : \mathbf{x} \in M\} \tag{28}$$

Let the *Boundary point* $\mathbf{y}^{Bound}$ of a set of objective vectors $\mathcal{Y}$ be the vector combining the worst value (i.e. largest value in case of a minimization objective) that occurs per objective:

$${y_i}^{Bound} = \max_{\mathbf{y} \in \mathcal{Y}} y_i, \forall i \in \{1,\ldots,w\} \tag{29}$$

Moreover, let the *Ideal point* of a set of objective vectors be the vector combining the best value (i.e. smallest value in case of a minimization objective) that occurs per objective:

$${y_i}^{Ideal} = \min_{\mathbf{y} \in \mathcal{Y}} y_i, \forall i \in \{1,\ldots,w\} \tag{30}$$

3.2 $\mathcal{S}$ Metric

3.2.1 Theory

The $\mathcal{S}$ metric was proposed by Zitler and Thiele [8], who called it the *size of the space covered*. Knowles and Corne [9] define it as calculating the hypervolume of the multi-dimensional region enclosed by the objective vectors $\mathcal{Y}_N$ and a reference point ($\mathbf{y}^{ref}$), hence computing the size of the region in objective space that M dominates. They point out that the choice of reference point is arbitrary (to an extent: the reference point should be

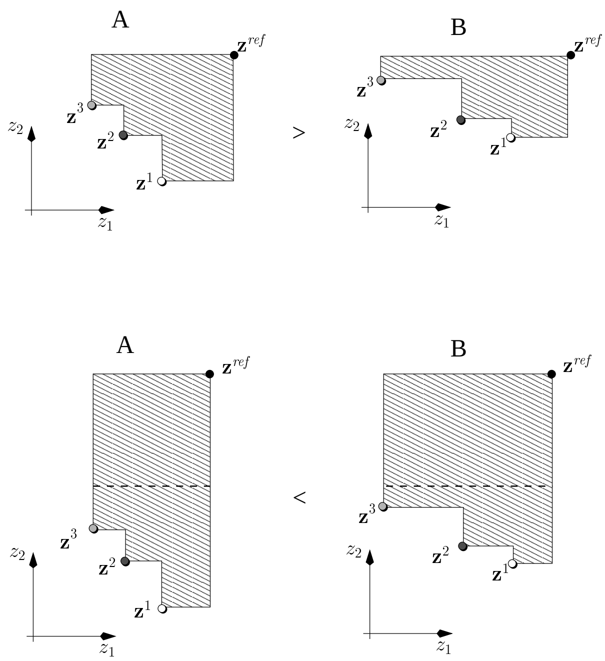


Figure 4: The relative value of the $\mathcal{S}$ metric depends upon the choice of reference point. In the upper half two non-dominated sets are shown, A and B, with $\mathcal{S}(A) > \mathcal{S}(B)$. In the lower half the same sets have a different ordering in $\mathcal{S}$. Please note: it is the objective space $\mathbb{Y}$ pictured, but Knowles and Corne [9] used a different notation $\mathbb{Z}$ ($z_i \leftrightarrow y_i$, $\mathbf{z} \leftrightarrow \mathbf{y}$). (Figure courtesy of Knowles and Corne [9].)

dominated by all objective vectors corresponding to decision vectors, i.e. $\boldsymbol{f}(\mathbf{x}) \prec \mathbf{y}^{ref} : \forall \mathbf{x} \in \mathbb{X}$) but still needs to be done carefully as it can effect the relative ordering of the compared non-dominated sets (see Figure 4 for their illustrated two dimensional case). Furthermore, an accurate calculation of the $\mathcal{S}$ metric requires a normalized and positive objective space [5].

Major pitfall of the $\mathcal{S}$ metric was the time complexity of the original algorithm calculating it, limiting its usefulness for large sets or problems with many objectives: $\mathcal{O}(k^{l+1})$, with k being the number of objective vectors in the non-dominated set and l being the number of objectives [9]. Better algorithms have been developed since; the current best result for $l = 3$ is $\mathcal{O}(k \log k)$ (by Beume et al. [11]) and the best time complexity for $l > 3$ is $\mathcal{O}(k \log k + k^{l/2})$ (by Beume and Rudolph [10]).

3.2.2 Implementation

In this study the three dimensional $\mathcal{S}$ metric algorithm by Emmerich [6] was used. While it was originally designed to determine the $\mathcal{S}$ contributions ($\Delta_{\mathcal{S}}$) of all objective vectors in a supplied non-dominated set, with some small augmentations it could be used for calculating the total $\mathcal{S}$ of a set as well (see Algorithm 2 in the Appendix). The algorithm operates in $\mathcal{O}(k^3)$ time[2].

In comparing the results of the NSGA-II and SMS-EMOA runs, normalization of the objective vectors was handled as follows:

$$y_i^{norm} = \frac{y_i - y_i^{IdealTP}}{y_i^{BoundTP} - y_i^{BoundTP}}, \forall i \in \{1, 2, 3\} \quad (31)$$

where:

$\mathbf{y}^{BoundTP}$	Boundary point of the test problem;
$\mathbf{y}^{IdealTP}$	Ideal point of the test problem.

Since the values for all objectives were already non-negative, this projects the objective space onto $[0, 1]^3$ and sets all objectives to be minimized.

The Boundary point of a test problem is obtained as follows (see Section 2.1.1, 2.1.2, 2.1.3, and 2.2 for used notation):

$$\mathbf{y}^{BoundTP} = \begin{bmatrix} F_1(\mathbf{D} = \{N^{diam} - 1\}^m) \\ F_2(S = 0) = 1 \\ F_3(\mathbf{Q}^{met} = \{0\}^n) = 0 \end{bmatrix} \quad (32)$$

[2]For calculating the total $\mathcal{S}$ metric of a set this algorithm is slower than the three dimensional algorithm by Beume et al. (mentioned in Section 3.2.1). When used in the SMS-EMOA algorithm however (see Section 3.5) the difference in computational speed is smaller since then all k $\Delta_{\mathcal{S}}$ contributions are needed; using the algorithm by Beume et al. k iterations would be required with a resulting time complexity of $\mathcal{O}(k^2 \log k)$.

where:

$$\mathbf{D} = (D_1, \ldots, D_m) \tag{33}$$

$$\mathbf{Q}^{met} = ({Q_{r+1}}^{met}, \ldots, {Q_{r+n}}^{met}) \tag{34}$$

Moreover, the Ideal point of a test problem is obtained as follows (see Section 2.1.1, 2.1.2, 2.1.3, and 2.2 for used notation):

$$\mathbf{y}^{IdealTP} = \begin{bmatrix} F_1(\mathbf{D} = \{0\}^m) \\ F_2(\mathbf{q} = \{1\}^m) \\ F_3(\mathbf{Q}^{met} = \mathbf{Q}) = 1 \end{bmatrix} \tag{35}$$

where:

$$\mathbf{q} = (q_1, \ldots, q_m) \tag{36}$$

$$\mathbf{Q} = (Q_{r+1}, \ldots, Q_{r+n}) \tag{37}$$

In comparing the results of the NSGA-II and SMS-EMOA runs, the following (effectively pre-normalized) reference point $\mathbf{y}^{ref}$ was used:

$${y_i}^{ref} = 1 + \varepsilon, \forall i \in \{1, 2, 3\} \tag{38}$$

where:

ε reference point constant, arbitrary non-negative value to make sure $[\boldsymbol{f}(\mathbf{x})]^{norm} \prec \mathbf{y}^{ref} : \forall \mathbf{x} \in \mathbb{X}$; in this study $\varepsilon = 0.001$ was used.

3.3 The General Evolutionary Algorithm

In order to do an efficient comparison between the definitions of NSGA-II and SMS-EMOA, the optimization loop of the general (multiobjective) evolutionary algorithm is laid out here (for an overview, see Algorithm 1). Certain assumptions (e.g. a fixed *population size*) are made, which could be argued not to hold for every evolutionary algorithm, but since they do for both NSGA-II and SMS-EMOA, we make them here. Initially, a random *parent population* of μ *individuals* (i.e. random decision vectors) is created. For these individuals values for each of the objectives are calculated (i.e. the individuals are evaluated which determines the associated objective vectors). Then the optimization loop starts, which continues until a certain *stop condition* is reached (often the number of decision vector *evaluations* is fixed, as these are likely to account for the greatest amount of time/money in real-life applications). The loop begins with selecting parents using a certain *selection* scheme, which will then be used in pairs or individually for reproduction by the *variation* operators. Variation creates λ *offspring*, using *recombination* followed by *mutation*, with probability $prob_{rec}$ and $prob_{mut}$ respectively. Offspring is evaluated, after which, using a second *selection* scheme, the μ best *ranked* individuals are taken from the combined parent and offspring pool to form the new parent population.

Section 3.4 and 3.5 address the NSGA-II and SMS-EMOA implementations of this general EA model; for the problem specific implementations of the EA operators, which are shared by NSGA-II and SMS-EMOA, see Section 4.

Algorithm 1 General EA

$t \leftarrow 0$
$P_t \leftarrow$ initialize() /* *Initialize random start population of μ individuals* */
evaluate(P_t) /* *Determine objective values for individuals in population* */
repeat
  $P'_t \leftarrow$ select-parents(P_t) /* *Select parent individuals that will be used for creating offspring* */
  $Q_t \leftarrow$ variation(P'_t) /* *Use recombination and mutation to create λ offspring* */
  evaluate(Q_t) /* *Determine objective values for offspring* */
  $P_{t+1} \leftarrow$ selection($P_t \cup Q_t$) /* *Select μ individuals that will form the new population* */
  $t \leftarrow t + 1$
until stop criterium reached

3.4 NSGA-II

The NSGA-II algorithm by Deb et al. [4] generates μ offspring per iteration (i.e. $\lambda = \mu$). The new parent population is selected using:

1. *Non-dominated sorting*: all individuals are assigned a non-domination rank, placing individuals with equal ranks in the same non-dominated front and with rank i indicating domination by individuals in $i - 1$ better ranked fronts (i.e. of $\{\mathcal{R}_1, \ldots, \mathcal{R}_I\}$, $\mathcal{R}_1$ is the actual Pareto front and points in the worst front $\mathcal{R}_I$ are dominated by points in $\{\mathcal{R}_1, \ldots, \mathcal{R}_{I-1}\}$) (see Figure 5 for a two dimensional example [6]).

2. *Crowding distance sorting*: for comparing individuals within the same front, individuals are assigned a second ranking indicating the distance between their nearest neighbors (i.e. at left- and righthand side, with larger distance leading to higher rank).

The new population is then composed as follows: add fronts, starting from rank 1, until the size of the current front exceeds the empty spaces remaining in the new population; then use the crowding distance ranking to fill the remaining space with the best individuals from this front. With complexity of the non-dominated sorting and crowding distance sorting being $\mathcal{O}(\mu^2)$ and $\mathcal{O}(\mu \log \mu)$ respectively, the complexity of NSGA-II is governed by the non-dominated sorting operation and is $\mathcal{O}(\mu^2)$ per iteration (for a three objective problem) [4].

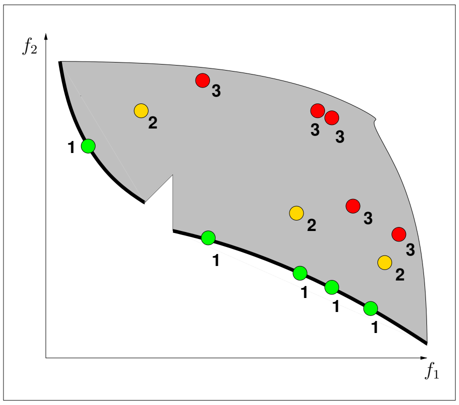


Figure 5: Illustration of the non-dominated sorting procedure for a population of 13 individuals in a two dimensional objective space. The population is divided into three partitions. The numbers attached to the solutions indicate the rank that was assigned to them by the non-dominated sorting procedure. (Figure courtesy of Emmerich [6].)

Formiga et al. [1] used the *MOMHLib++* [12] implementation of Deb's algorithm, an approach that was followed in this study in order to allow for a comparison between test results. It is a genuine implementation, except for the fact that instead of the final parent population an unbounded archive is presented as test result, containing all mutually non-dominated individuals generated over all iterations of the algorithm (much like the way an archive is maintained in *PEAS* by Knowles and Corne, which approach Deb et al. describe in [4], but a concept they did not include in the definition of NSGA-II). For this study the MOMHLib++ implementation was adapted so not the unbounded archive but the final population is taken as test result.

3.5 SMS-EMOA

The SMS-EMOA algorithm by Emmerich et al. [5] generates 1 offspring per iteration (i.e. $\lambda = 1$). The new parent population is selected using:

1. *Non-dominated sorting*: see Section 3.4.
2. $\mathcal{S}$ *metric contribution sorting*: for comparing individuals within the same front, individuals are assigned a second ranking indicating their $\Delta_{\mathcal{S}}$ with regard to the front in which they are contained (see Section 3.2.2).

The new population is determined as follows: non-dominated sorting is used to divide the individuals over fronts and the worst (i.e. last) front is then sorted on $\Delta_{\mathcal{S}}$. The individual in the last front with the lowest $\Delta_{\mathcal{S}}$ is discarded, which leaves μ individuals in the combined parent and offspring pool

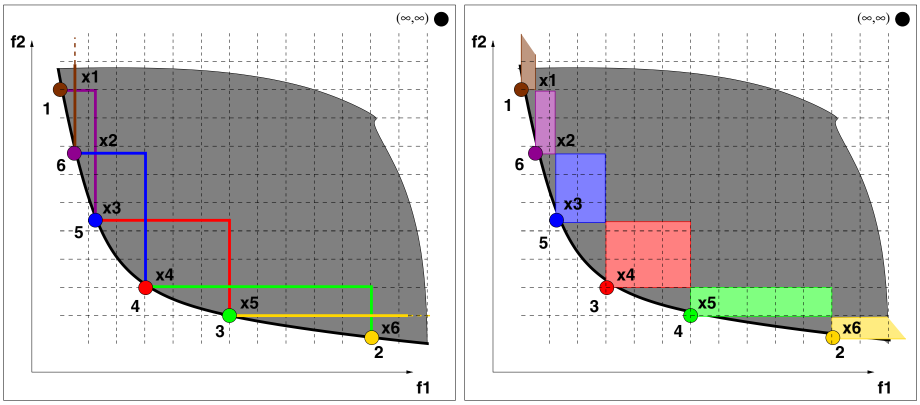


Figure 6: Comparison of crowding distance sorting (left) and sorting by $\Delta_{\mathcal{S}}$ (right). Concerning the inner points of the front, x5 (rank 3) outperforms x4 (rank 4), if the crowding distance is used as a ranking criterion. On the other hand, x4 (rank 3) outperforms x5 (rank 4), if $\Delta_{\mathcal{S}}$ is employed. This indicates that good compromise solutions, which are located near knee-points of convex parts of the Pareto front, are given better ranks in SMS-EMOA than in the NSGA-II algorithm. (Figure courtesy of Emmerich et al. [5].)

for forming the new parent population. With complexity of non-dominated sorting (which was borrowed from NSGA-II) and $\mathcal{S}$ metric contribution sorting being $\mathcal{O}(\mu^2)$ ([4]) and $\mathcal{O}(\mu^3)$ respectively, the complexity of SMS-EMOA is governed by the $\mathcal{S}$ metric contribution sorting operation and is $\mathcal{O}(\mu^3)$ per iteration (for a three objective problem, using the algorithm by Emmerich [6] for the $\mathcal{S}$ metric contribution sorting (see Section 3.2.2)).

For determining the $\Delta_{\mathcal{S}}$ of individuals in the last front, normalization is applied and a *dynamic* reference point is used (as proposed by Emmerich et al. [5]). The approach is similar to the procedure for comparing results of NSGA-II and SMS-EMOA runs (see Section 3.2.2), except for the fact $\mathbf{y}^{Bound\mathcal{R}_I}$ and $\mathbf{y}^{Ideal\mathcal{R}_I}$ are used instead of $\mathbf{y}^{BoundTP}$ and $\mathbf{y}^{IdealTP}$ (i.e. the Boundary and Ideal points of the last front $\mathcal{R}_I$ are taken, instead of those of the test problem).

The most important difference between SMS-EMOA and NSGA-II is the different sorting by the $\mathcal{S}$ metric contribution and crowding distance operations: see Figure 6 for a two dimensional example [5].

4 Problem Specific Approach

A problem specific approach was devised, specifically tailored for the problem of multiobjective optimization of water distribution networks and for usage in both the NSGA-II and SMS-EMOA implementations. This way, differences in the test results can with certainty be attributed to the differences between the algorithms. Firstly the *individual* (or *solution*) *representation* is presented, then the *operators* are addressed and lastly an overview of the *parameters* is given that were used in the different test runs. See Section 2.1.1, 2.1.2, and 2.1.3 for used notation.

4.1 Representation

An individual is represented by an m-tuple containing a diameter code for each pipe:

$$(D_1, \ldots, D_m) \tag{39}$$

with:

$$D_i \in \{0, 1, \ldots, N^{diam} - 1\}, \forall i \in \{1, \ldots, m\} \tag{40}$$

4.2 Operators

The selection operator for forming a new population P_{t+1}, which is done differently for NSGA-II and SMS-EMOA, is described in Section 3.4 and 3.5. The remaining operators are defined here.

4.2.1 Initialization

The initial population is filled with μ uniform randomly initialized individuals, i.e.:

$$D_i = u \in \{0, 1, \ldots, N^{diam} - 1\}, \forall i \in \{1, \ldots, m\} \tag{41}$$

where:

u uniform randomly picked integer.

4.2.2 Parent Selection

Parent individuals are selected from the parent population P_t using *Binary Tournament* selection; this is done twice per iteration for obtaining two parent individuals. The procedure is as follows: uniform randomly pick two individuals from the parent population P_t and decide a winner by comparing the front in which they are contained (best front wins); if they are both in the same front, decide by crowding distance (NSGA-II; largest value wins) or use individual 2 (SMS-EMOA; it is too costly to determine $\Delta_{\mathcal{S}}$ for individuals in all fronts).

4.2.3 Recombination

Recombination is applied with probability $prob_{rec}$ (see Section 4.3). The two parent individuals, delivered by the parent selection operator, are combined to form one offspring individual using *uniform crossover*. The procedure is as follows: starting from two parent individuals, diameter code D_i is copied from parent 1 to the new offspring individual with probability 0.5; it is taken from parent 2 otherwise. Repeat for all $i \in \{1, \ldots, m\}$. In case of the recombination step not being taken, parent 1 is simply copied as offspring.

4.2.4 Mutation

Mutation is applied on each diameter code D_i of the newly created offspring with probability $prob_{mut}$ (see Section 4.3). A new value is chosen for diameter code D_i:

$$D_i = u \in \{0, 1, \ldots, N^{diam} - 1\} \tag{42}$$

where:

u uniform randomly picked integer.

4.3 Parameters

Per test problem three test setups with different parameters (see Table 1) have been run, using our NSGA-II and SMS-EMOA implementation. The first setup is based on the settings Formiga et al. [1] used (i.e. $prob_{rec} = 1$, $prob_{mut} = 0.05$), setup 2 and 3 are included to study the effect of using only mutation as variation operator, and with setup 4 and 5, the effect of a smaller population size is assessed. All test setups have 20,000 evaluations by the hydraulic simulator as stop condition, which allows the algorithms to generate 20,000 individuals.

Table 1: Test Setups

Setup	μ	$\boldsymbol{prob_{rec}}$	$\boldsymbol{prob_{mut}}$
1	100	1	0.05
2	100	0	0.05
3	100	0	$\frac{1}{m}$
4	50	0	0.05
5	50	0	$\frac{1}{m}$

5 Test Problems

In literature a recurring set of test problems is used, from which we chose *Two Loop* and *Hanoi* to apply optimization to (following Formiga et al. [1]). Two Loop is a fictional network, originally defined in [13] by Alperovits and Shamir (1977), and has a relatively low complexity. Hanoi is the water distribution trunk layout that was to be realized in Vietnam, originally defined in [14] by Fujiwara and Khang (1990), and being larger than Two Loop (i.e. more network components like pipes, nodes, and loops), it has a higher complexity. Both problems deal with the design of a new water distribution

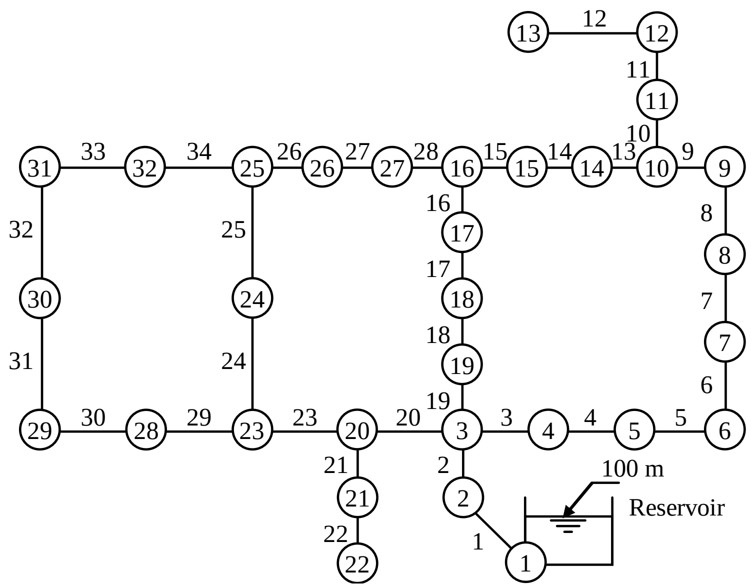


Figure 7: The Hanoi Network. (Figure courtesy of Tospornsampan et al. [15].)

network, but different more complex systems, like the New York City parallel expansion problem [14] (i.e. placement of new pipes parallel to existing ones, in order to service an increased demand), are found in literature.

5.1 Two Loop

The Two Loop network (see Figure 2) has 8 pipes, 7 nodes and one reservoir (see Table 6 and 7, Appendix). No pumping facilities are used, the internal nodes are fed by gravity. All pipes have roughness coefficient $C_i = 130$ for usage in the Hazen-Williams equation (see Section 2.1.5). 14 commercial diameters are available (see Table 8, Appendix), p^{zero} is taken to be 10m [1] and p^{req} is 30m (see Section 2.1.1 for used notation).

5.2 Hanoi

The Hanoi network (see Figure 7) has 34 pipes, 31 nodes and one reservoir (see Table 9 and 10, Appendix). No pumping facilities are used, the internal nodes are fed by gravity. All pipes have roughness coefficient $C_i = 130$ for usage in the Hazen-Williams equation (see Section 2.1.5). The original problem included 6 commercial diameters, but this set was found to be insufficient by Formiga et al. [1], so they added 4 larger diameters (see Table 11, Appendix). p^{zero} is taken to be 15m [1] and p^{req} is 30m (see Section 2.1.1 for used notation).

6 Results

The results of the test runs using SMS-EMOA and NSGA-II, with setups 1 through 5 (see Table 1) and applied to problems Two Loop and Hanoi, can be found in Table 2 and 3. In the following sections, these results are analyzed by comparing the solutions of Formiga's NSGA-II implementation (NSGA-II-F) with the outcome of the NSGA-II implementation that was used in this study (NSGA-II-R), by comparing the NSGA-II results with the SMS-EMOA results, and by comparing the test setups themselves while generalizing over the SMS-EMOA and NSGA-II results.

Table 2: Test Results for Two Loop

Setup	Algorithm	$\mathcal{S}$ metric	Rank	Total
1:	NSGA-II	0.700108	4	9
$(100, 1, 0.05)$	SMS-EMOA	0.763592	2	2
2:	NSGA-II	0.743674	2	7
$(100, 0, 0.05)$	SMS-EMOA	0.763182	3	3
3:	NSGA-II	0.761553	1	4
$(100, 0, \frac{1}{m})$	SMS-EMOA	**0.764674**	1	1
4:	NSGA-II	0.659611	5	10
$(50, 0, 0.05)$	SMS-EMOA	0.760382	5	6
5:	NSGA-II	0.741382	3	8
$(50, 0, \frac{1}{m})$	SMS-EMOA	0.760698	4	5

Table 3: Test Results for Hanoi

Setup	Algorithm	$\mathcal{S}$ metric	Rank	Total
1:	NSGA-II	0.58961	1	3
$(100, 1, 0.05)$	SMS-EMOA	0.486277	4	9
2:	NSGA-II	0.58175	2	4
$(100, 0, 0.05)$	SMS-EMOA	0.564249	3	6
3:	NSGA-II	0.577036	3	5
$(100, 0, \frac{1}{m})$	SMS-EMOA	**0.601679**	1	1
4:	NSGA-II	0.559779	4	7
$(50, 0, 0.05)$	SMS-EMOA	0.425922	5	10
5:	NSGA-II	0.529533	5	8
$(50, 0, \frac{1}{m})$	SMS-EMOA	0.591234	2	2

6.1 NSGA-II-F vs. NSGA-II-R

In order to make an efficient comparison possible, $\mathcal{S}$ metric scores were determined for the test results by Formiga et al. [1] (see Table 4). Per test problem the best NSGA-II-F and the best NSGA-II-R set was selected for a graphical comparison (see Figure 8 and 9). Both selected NSGA-II-F setups

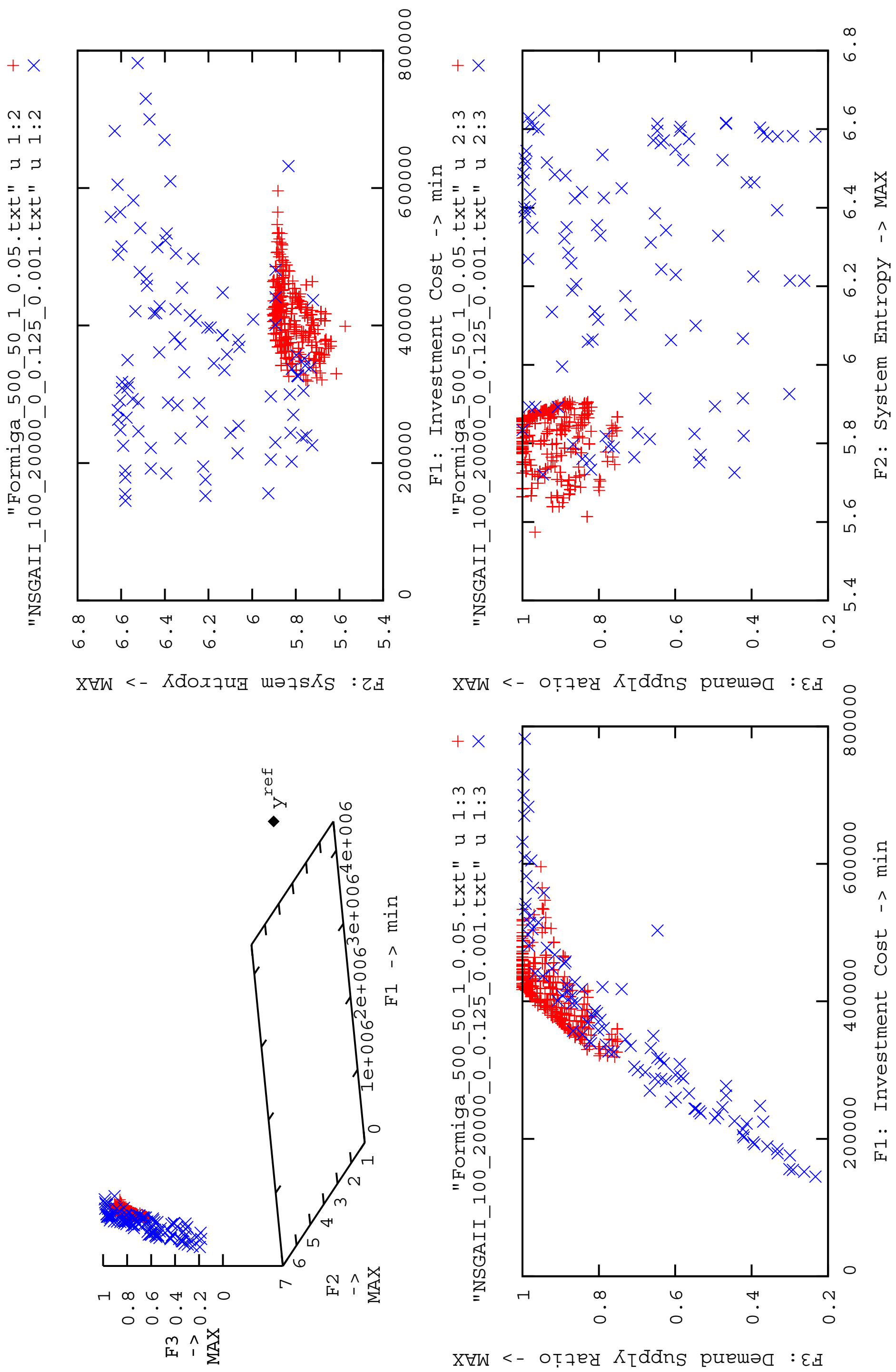


Figure 8: NSGA-II Comparison for Two Loop. The best result by Formiga et al. [1] ($\mathcal{S} = 0.650451$, setup 8: $\mu = 500$, 50 generations, $prob_{rec} = 1$, $prob_{mut} = 0.05$; see Table 4) and the best NSGA-II result of this study ($\mathcal{S} = 0.761553$, setup 3: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 0$, $prob_{mut} = \frac{1}{m}$; see Table 2) are shown.

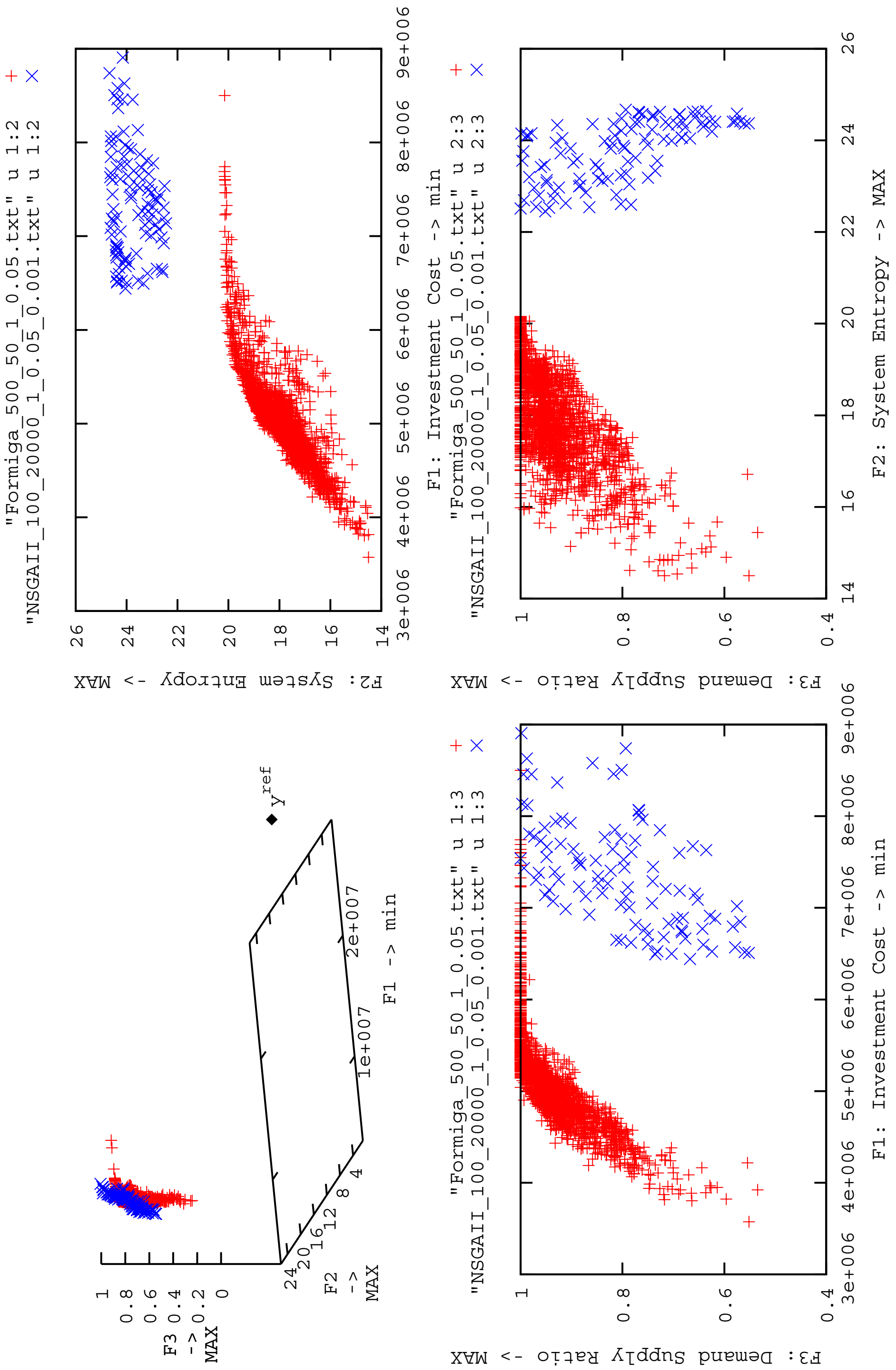


Figure 9: NSGA-II Comparison for Hanoi. The best result by Formiga et al. [1] ($\mathcal{S} = 0.531216$, setup 8: $\mu = 500$, 50 generations, $prob_{rec} = 1$, $prob_{mut} = 0.05$; see Table 4) and the best NSGA-II result of this study ($\mathcal{S} = 0.58961$, setup 1: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 1$, $prob_{mut} = 0.05$; see Table 3) are shown.

used $\mu = 500$ and 50 generations resulting in 25,000 generated solutions (i.e. evaluations by the hydraulic simulator) in total, whereas in this study for all test runs a cap of 20,000 evaluations was used.

For Two Loop the NSGA-II-R set has the higher $\mathcal{S}$ metric score (0.761553 vs. 0.650451, see Table 2 and 4). Furthermore on the one hand the NSGA-II-R set has higher values for System Entropy, but on the other hand the NSGA-II-F set has a higher overall value for Demand Supply Ratio. Moreover it is striking to see that the NSGA-II-R set has a much higher diversity in its solutions than the NSGA-II-F set, probably because of implicit constraints inherent to the hydraulic simulator Formiga et al. used[3].

For Hanoi the NSGA-II-R again has the higher $\mathcal{S}$ metric score, although the difference is smaller this time (0.58961 vs. 0.531216, see Table 3 and 4). While the NSGA-II-R solutions are more expensive than their NSGA-II-F counterparts, this apparently enables a gain in System Entropy. The higher $\mathcal{S}$ metric score is likely to be attributed to this combination of high System Entropy against higher Investment Cost, as the values for Demand Supply Ratio appear equally spread for both sets.

Table 4: $\mathcal{S}$ Metric Scores for Test Results by Formiga et al. [1]

			Two Loop		**Hanoi**	
Setup	μ	**Generations**	$\mathcal{S}$ **metric**	**Rank**	$\mathcal{S}$ **metric**	**Rank**
1	50	50	0.648698	5	0.506338	9
2	50	100	0.647552	7	0.520825	7
3	100	50	0.648642	6	0.524329	6
4	100	100	0.644359	8	0.519355	8
5	250	50	0.648727	4	0.528566	4
6	250	100	0.650414	2	0.529112	2
7	250	200	n.a.	n.a.	0.525984	5
8	500	50	**0.650451**	1	**0.531216**	1
9	500	100	0.64996	3	0.5290460	3

Table 5: Simulators Compared: $y_i^{Reehuis} - y_i^{Formiga}$

	Two Loop		**Hanoi**	
i	**mean**	**norm.abs.**	**mean**	**norm.abs.**
1: min	0	0	+19.1785	$7.64917 \cdot 10^{-7}$
2: max	$-1.22527 \cdot 10^{-3}$	$2.44092 \cdot 10^{-3}$	+1.16954	0.0355562
3: max	$-1.43839 \cdot 10^{-3}$	0.0171752	−0.404399	0.404399

It should be noted that there are some limitations to the reliability of this Formiga vs. Reehuis comparison. These occur firstly because of the dif-

[3]As Dr. Formiga stated over email: generated solutions that could not be evaluated by the hydraulic simulator in a timely fashion (i.e. that did not have convergence in 100 iterations of the hydraulic simulator) were penalized and removed.

ferences in calculated objective values that the different hydraulic simulators (in combination with the different objective function implementations) give rise to. The solutions in the sets by Formiga et al. that were selected for this comparison (depicted in Figure 8 and 9) were fed to the hydraulic simulator used in this study in order to compare calculated objective values, see Table 5. For Two Loop the differences are neglectable, but for Hanoi Formiga et al. obtained significantly higher values for Demand Supply Ratio. Secondly, as Formiga et al. did not use the final population sets but unbounded archives as their test results (see Section 3.4), their sets contain (much) more solutions. One could argue however that with the highest scores going to the NSGA-II-R sets and both these factors being favorable to the results by Formiga et al., they are of less consequence.

6.2 NSGA-II vs. SMS-EMOA

When examining the $\mathcal{S}$ metric scores for Two Loop in Table 2 it is clear that the SMS-EMOA results outrank the NSGA-II results; the 4^{th} and 7^{th} are the highest NSGA-II rankings. For a graphical comparison of the best NSGA-II set and SMS-EMOA set for Two Loop, see Figure 10. Both sets show an equal spread for Investment Cost and Demand Supply Ratio, but the SMS-EMOA set has a higher overall value for System Entropy.

Examining the Hanoi scores in Table 3, SMS-EMOA again scores better than NSGA-II, but the outcome is less convincing than for Two Loop. SMS-EMOA holds the 1^{th} and 2^{nd} rank, but the lowest ranks 9 and 10 are also held by SMS-EMOA results. Moreover, these worst SMS-EMOA sets are of such poor quality that for Hanoi the mean SMS-EMOA score drops below the mean NSGA-II score: 0.5338722 vs. 0.5675416. For a graphical comparison of the best NSGA-II set and SMS-EMOA set for Hanoi, see Figure 11. The SMS-EMOA set contains solutions combining both low Investment Cost and high System Entropy, while the NSGA-II set has higher overall Demand Supply Ratio.

For an illustration of the $\mathcal{S}$ development of the best NSGA-II and SMS-EMOA sets per test problem, see Figure 12 in the Appendix.

6.3 Setups Compared

When generalizing over the SMS-EMOA results (see Table 2 and 3), usage of the following can be stated to increase the $\mathcal{S}$ score:

1. Larger population size ($\mu = 100$);
2. Mutation rate with probability of changing one diameter code D_i ($prob_{mut} = \frac{1}{m}$).

On the other hand, when generalizing over the NSGA-II results (see Table 2 and 3), usage of the following can be stated to increase the $\mathcal{S}$ score:

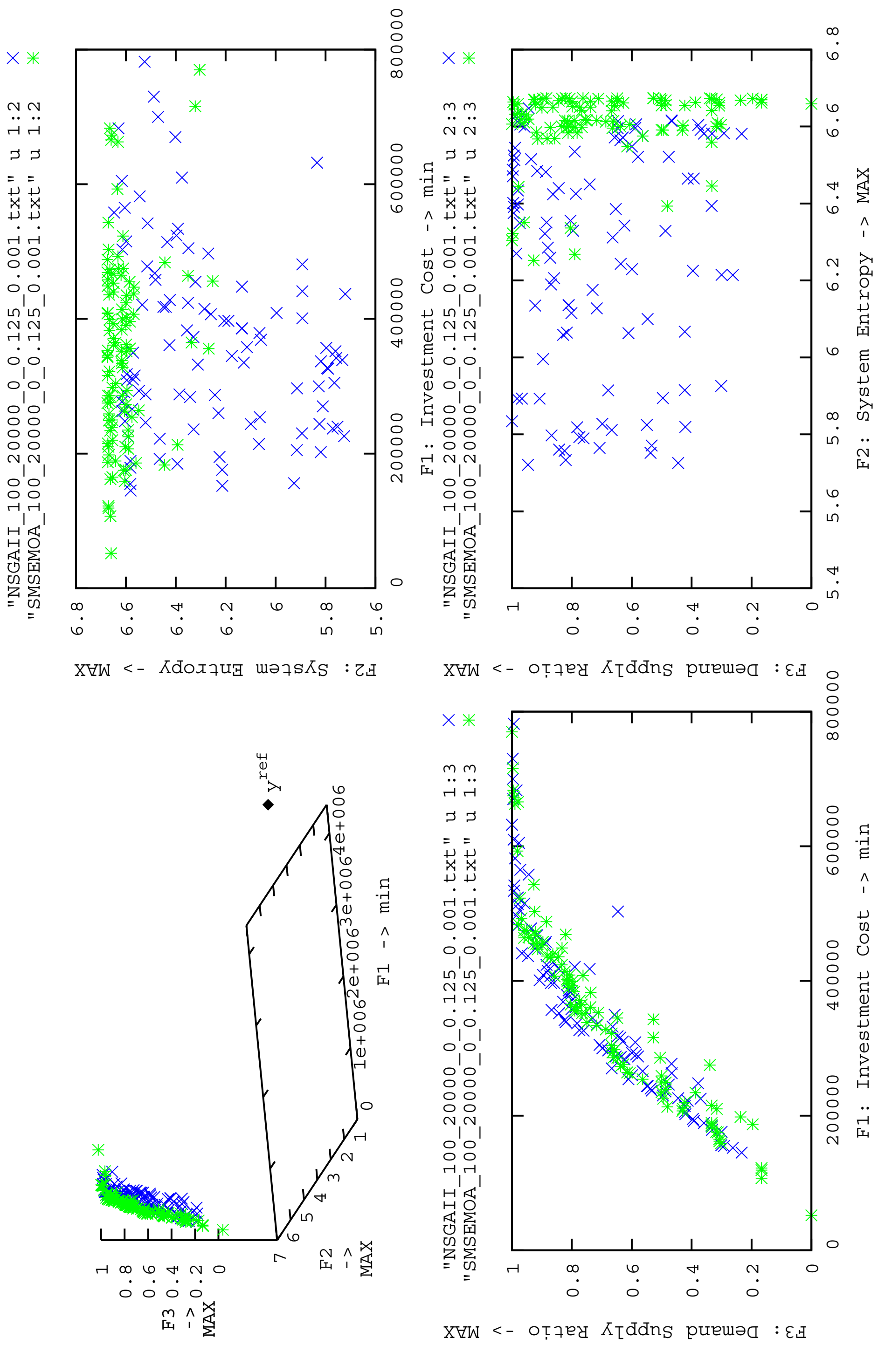


Figure 10: NSGA-II vs. SMS-EMOA for Two Loop. The best NSGA-II result ($\mathcal{S} = 0.761553$, setup 3: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 0$, $prob_{mut} = \frac{1}{m}$) and the best SMS-EMOA result ($\mathcal{S} = 0.764674$, setup 3: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 0$, $prob_{mut} = \frac{1}{m}$) are shown; see Table 2.

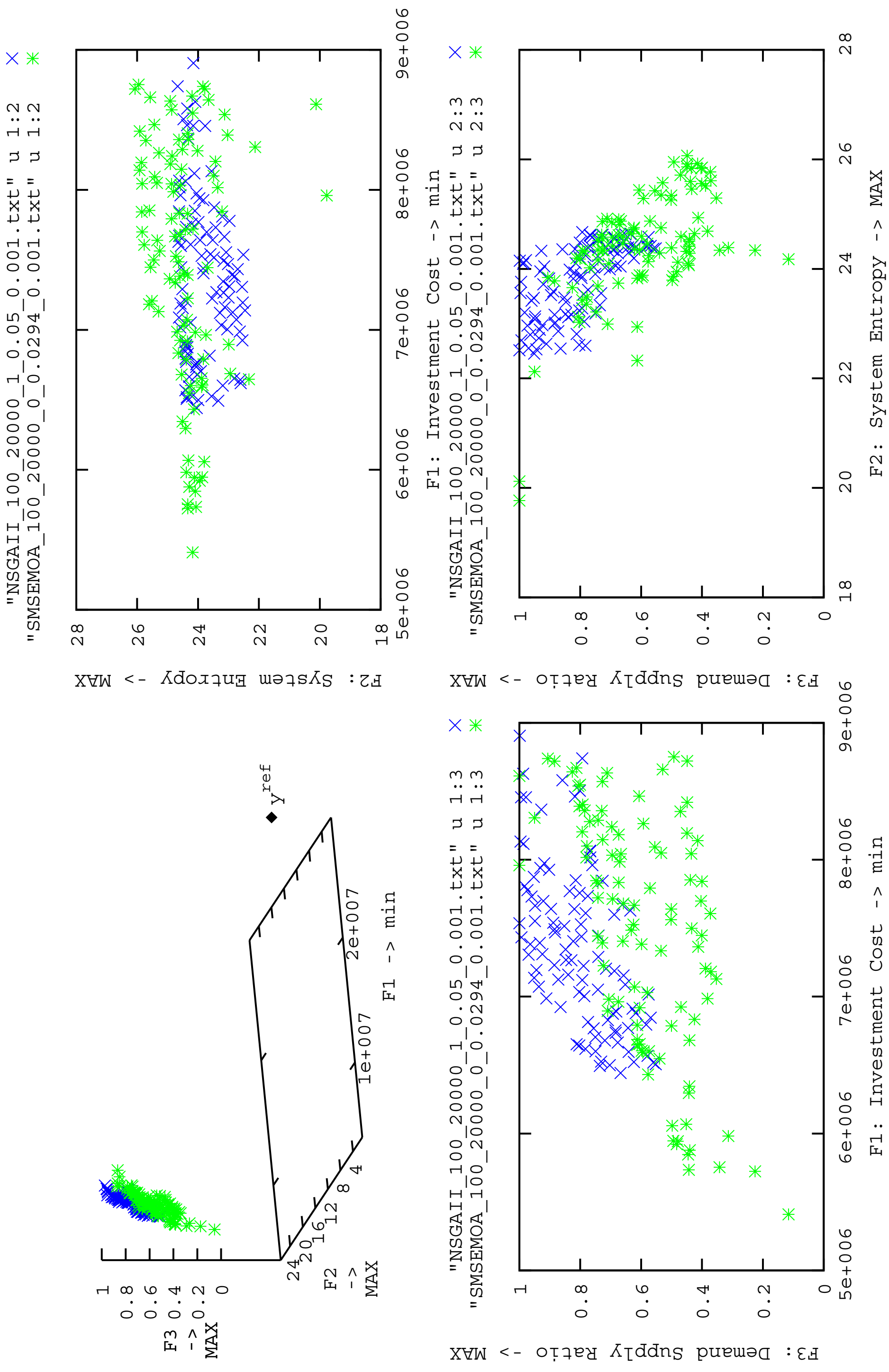


Figure 11: NSGA-II vs. SMS-EMOA for Hanoi. The best NSGA-II result ($\mathcal{S} = 0.58961$, setup 1: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 1$, $prob_{mut} = 0.05$) and the best SMS-EMOA result ($\mathcal{S} = 0.601679$, setup 3: $\mu = 100$, 20,000 evaluations, $prob_{rec} = 0$, $prob_{mut} = \frac{1}{m}$) are shown; see Table 3.

1. Larger population size ($\mu = 100$);
2. Larger mutation rate (Two Loop: $prob_{mut} = \frac{1}{m} > 0.05$, Hanoi: $prob_{mut} = 0.05 > \frac{1}{34} \approx 0.0294$);

Whether leaving out recombination is beneficial for the $\mathcal{S}$ score did not become clear, as for both SMS-EMOA and NSGA-II none of the setups 1 and 2 gets a better ranking over the other for both test problems. Futhermore, although a smaller population size degrades the final result, it seems to preserve the ordering determined by the other parameters. In practice this means one could save time needed for doing experiments by first varying the other parameters while using a small population size and then examining the best value for population size independently.

For the three dimensional SMS-EMOA this is an attractive property, since the decrease in time complexity when using a population of half its original size is substantial[4]. Comparing the complexity of one iteration (see Section 3.5) for $\mu = 100$ and $\mu = 50$, we get:

$$\mathcal{O}(\mu^3) \Rightarrow \left(\frac{100}{50}\right)^3 = 8 \tag{43}$$

This holds for NSGA-II as well, but is of less consequence since its time complexity is lower to begin with, which also makes the decrease less substantial. Again, comparing the complexity of one iteration (see Section 3.4) for $\mu = 100$ and $\mu = 50$:

$$\mathcal{O}(\mu^2) \Rightarrow \left(\frac{100}{50}\right)^2 = 4 \tag{44}$$

7 Conclusions

The goal of this study was to determine which of the multiobjective evolutionary algorithms NSGA-II [4] and SMS-EMOA [5] would provide the best non-dominated set or Pareto front for the problem of multiobjective optimization of water distribution networks, using the $\mathcal{S}$ metric as comparison criterion. Based on the results in Table 2 and 3, it is clear NSGA-II does not outperform SMS-EMOA. On the other hand, SMS-EMOA cannot be appointed the real winner either. Although it clearly has superior scores for test problem Two Loop, the results for the more complex Hanoi test problem are ambiguous: SMS-EMOA delivered the best two sets, but also

[4]For this study the experiments were carried out on an AMD Athlon64 X2 1.80 GHz machine with 2,00 GB of memory, running a 32-bit version of Windows Vista, and using a single core C++ compiler. On average, the SMS-EMOA runs took 2h6m (Two Loop, $\mu = 100$), 24m (Two Loop, $\mu = 50$), 3h15m (Hanoi, $\mu = 100$), and 39m (Hanoi, $\mu = 50$). The average NSGA-II run took 30s (Two Loop, $\mu = 100$), 20s (Two Loop, $\mu = 50$), 2m3s (Hanoi, $\mu = 100$), and 1m35s (Hanoi, $\mu = 50$).

generated the sets holding the lowest two ranks. Furthermore, this makes that for Hanoi the mean SMS-EMOA $\mathcal{S}$ metric score is lower than the mean NSGA-II $\mathcal{S}$ metric score: 0.5338722 vs. 0.5675416.

Analyzing the plots in Figure 10 and 11, they show that both pictured SMS-EMOA sets are higher on overall System Entropy than their NSGA-II counterparts. For Two Loop the sets of both algorithms show an equal spread for Investment Cost and Demand Supply Ratio; for Hanoi SMS-EMOA includes some cheaper solutions, but NSGA-II then shows higher overall Demand Supply Ratio.

What can be concluded is that SMS-EMOA holds the potential of outperforming NSGA-II in this field, provided the right set of parameters is used. In a future study the emphasis should lie on separately determining the best set of parameters for SMS-EMOA and for NSGA-II. This tuning should be done on the examined test problems Two Loop and Hanoi, but also on more complex systems like the New York City parallel expansion problem[5] [14]. Multiple runs using the same parameter set, but with different seeds for the random number generator[6], should be executed in order to determine possibly significant variation in performance per algorithm.

Acknowledgements

The author would like to thank Dr. Klebber Formiga for providing the test results of his study and his instant responses to questions. Special thanks go out to the supervisors of this project, Michael Emmerich and André Deutz, for their overall support and the lengthy discussions.

References

[1] Formiga, K.T.M., Chaudhry, F., Cheung, B., Reis, L.: Optimal Design of Water Distribution System by Multiobjective Evolutionary Methods. *Evolutionary Multi-Criterion Optimization*, Springer, 2003, LNCS **2632**, 677-691

[2] Rossman, L.A.: EPANET 2 Users Manual. United States Environmental Protection Agency, 2000, Water Supply and Water Resources

[5]For examining a parallel expansion problem certain adjustments are necessary: a diameter couple with size and price zero needs to be added to the table of commercial diameters, each pipe gets assigned an existing diameter, and for each pipe a copy is added which connects to same nodes and has initial diameter zero. The optimization algorithm should then be adapted so it initializes its solutions using the existing diameters and is only allowed to change the diameters of the second set of pipes.

[6]The random number generator is used in the SMS-EMOA and NSGA-II implementations for generating new solutions and applying the variation operators (i.e. recombination and mutation).

Division, National Risk Management Research Laboratory
http://www.epanet.com

[3] Savic, D.A., Walters, G.A.: Genetic Algorithms for Least-Cost Design of Water Distribution Networks. *Journal of Water Resources Planning and Management*, 1997, **123**(2), 67-77

[4] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. *Parallel Problem Solving from Nature PPSN VI*, Springer, 2000, LNCS **1917**, 849-858

[5] Emmerich, M., Beume, N., Naujoks, B.: An EMO Algorithm Using the Hypervolume Measure as Selection Criterion. *Evolutionary Multi-Criterion Optimization*, Springer, 2005, LNCS **3410**, 62-76

[6] Emmerich, M.T.M.: Single- and Multi-objective Evolutionary Design Optimization Assisted by Gaussian Random Field Metamodels. Dissertation for Doctoral Degree in Natural Sciences, Informatik, University of Dortmund, 2005

[7] Naujoks, B., Beume, N., Emmerich, M.: Multi-objective Optimisation using S-metric Selection: Application to three-dimensional Solution Spaces. *Proceedings of the 2005 Congress on Evolutionary Computation (CEC '05)*, IEEE, 2005, 1282-1289

[8] Zitzler, E., Thiele, L.: Multiobjective Optimization Using Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 1999, **3**(4), 257-271

[9] Knowles, J., Corne, D.: On Metrics for Comparing Nondominated Sets. *Proceedings of the 2002 Congress on Evolutionary Computation (CEC '02)*, IEEE, 2002, **1**, 711-716

[10] Beume, N., Rudolph, G.: Faster S-Metric Calculation by Considering Dominated Hypervolume as Klee's Measure Problem. *Proceedings of the Second IASTED Conference on Computational Intelligence*, ACTA Press, 2006, 231-236

[11] Beume, N., Fonseca, C.M., Lpez-Ibez, M., Paquete, L., Vahrenhold, J.: On the Complexity of Computing the Hypervolume Indicator. Reihe CI 235/07, SFB 531, University of Dortmund, 2007

[12] Jaszkiewicz, A.: Multiobjective Methods Metaheuristic Library for C++. http://www-idss.cs.put.poznan.pl/~jaszkiewicz/MOMHLib

[13] Alperovits, E., Shamir, U.: Design of Optimal Water Distribution Systems. *Water Resources Research*, 1977, **13**(6), 885-900

[14] Fujiwara, O., Khang, D.B.: A Two-Phase Decomposition Method for Optimal Design of Looped Water Distribution Networks. *Water Resources Research*, 1990, **26**(4), 539-549

[15] Tospornsampan, J., Kita, I., Ishii, M., Kitamura, Y.: Split-Pipe Design of Water Distribution Networks Using a Combination of Tabu Search and Genetic Algorithm. *International Journal of Computer, Information, and Systems Science, and Engineering*, WASET, 2007, **1**(3), 164-174

Appendix

Table 6: Node Data for Two Loop [13]

Node	**Elevation (meter)**	**Demand (m^3/hr)**
1	210	(reservoir)
2	150	100
3	160	100
4	155	120
5	150	270
6	165	330
7	160	200

Table 7: Pipe Data for Two Loop [13]

Pipe	b_i	e_i	**Length (meter)**
1	1	2	1000
2	2	3	1000
3	2	4	1000
4	4	5	1000
5	4	6	1000
6	6	7	1000
7	3	5	1000
8	5	7	1000

Table 8: Diameters for Two Loop [13]

Code	Size (inch)	Price (units/meter)
0	1	2
1	2	5
2	3	8
3	4	11
4	6	16
5	8	23
6	10	32
7	12	50
8	14	60
9	16	90
10	18	130
11	20	170
12	22	300
13	24	550

Table 9: Node Data for Hanoi [14]

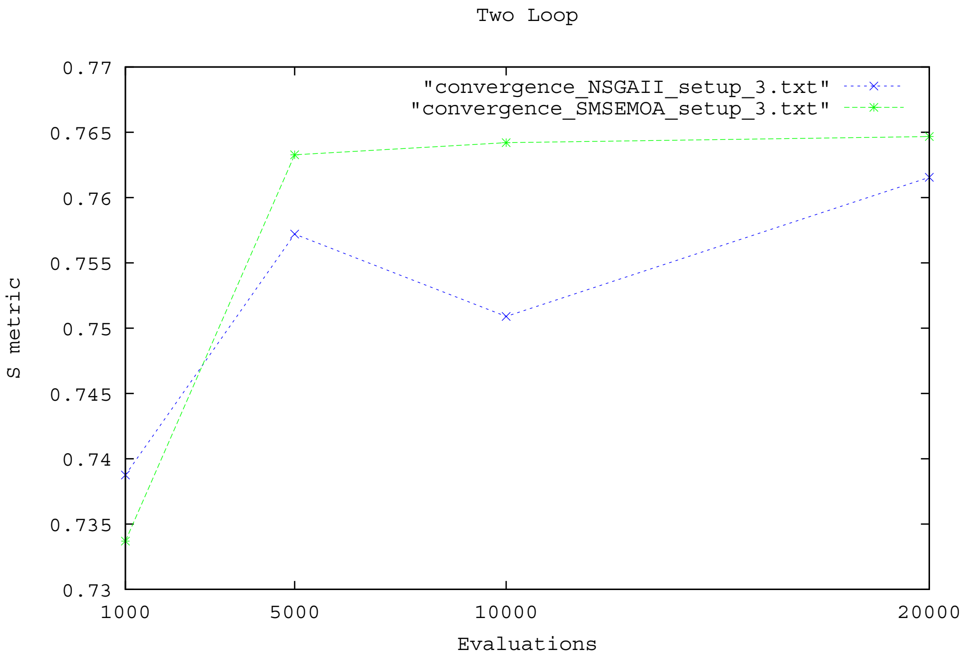
Node	**Elevation (meter)**	**Demand (m^3/hr)**
1	100	(reservoir)
2	0	890
3	0	850
4	0	130
5	0	725
6	0	1005
7	0	1350
8	0	550
9	0	525
10	0	525
11	0	500
12	0	560
13	0	940
14	0	615
15	0	280
16	0	310
17	0	865
18	0	1345
19	0	60
20	0	1275
21	0	930
22	0	485
23	0	1045
24	0	820
25	0	170
26	0	900
27	0	370
28	0	290
29	0	360
30	0	360
31	0	105
32	0	805

Table 10: Pipe Data for Hanoi [14]

Pipe	b_i	e_i	**Length (meter)**
1	1	2	100
2	2	3	1350
3	3	4	900
4	4	5	1150
5	5	6	1450
6	6	7	450
7	7	8	850
8	8	9	850
9	9	10	800
10	10	11	950
11	11	12	1200
12	12	13	3500
13	13	14	800
14	14	15	500
15	15	16	550
16	17	16	2730
17	18	17	1750
18	19	18	800
19	3	19	400
20	3	20	2200
21	20	21	1500
22	21	22	500
23	20	23	2650
24	23	24	1230
25	24	25	1300
26	26	25	850
27	27	26	300
28	16	27	750
29	23	28	1500
30	28	29	200
31	29	30	1600
32	30	31	150
33	32	31	860
34	25	32	950

Table 11: Diameters for Hanoi [14], [1]

Code	Size (inch)	Price (units/meter)
0	12	45.73
1	16	70.40
2	20	98.39
3	24	129.33
4	30	180.75
5	40	278.28
6	50	388.91
7	60	511.23
8	70	644.23
9	80	787.10



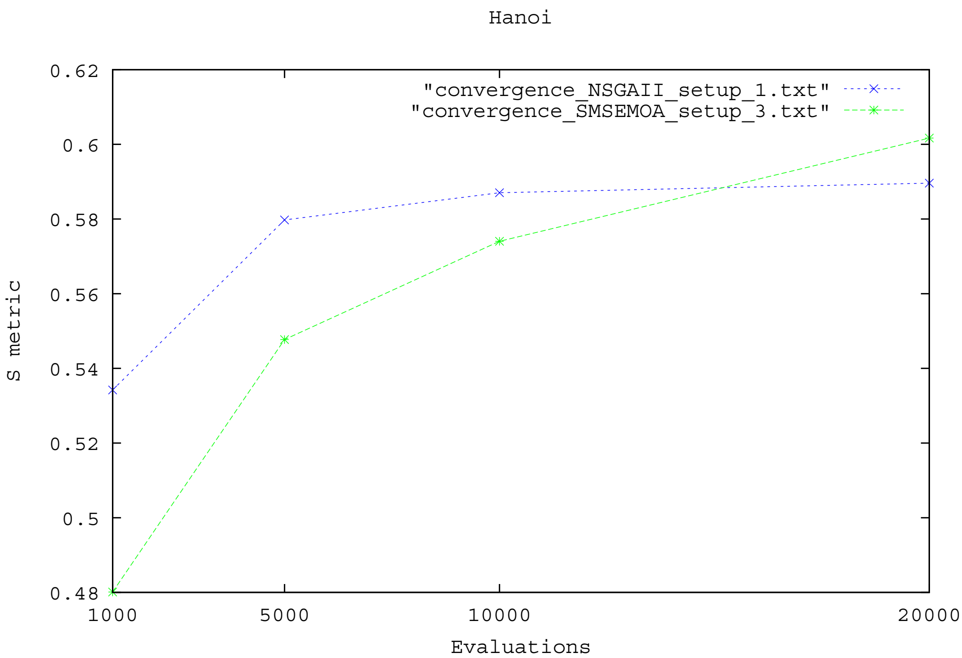


Figure 12: Convergence for Two Loop and Hanoi. Per test problem the convergence of the best run of NSGA-II (Two Loop: setup 3, Hanoi: setup 1) and of SMS-EMOA (Two Loop: setup 3, Hanoi: setup 3) is shown (see Table 2 and 3).

Algorithm 2 $\mathcal{S}$ METRIC_3D(*total*)

```
if (total = true) then
   front ← R_1                      /* Use first front for determining total S */
else
   front ← R_I                      /* Use last front for determining Δ_S */
end if
k ← |front|
/* Part 1: Initialization */
values-f_1 ← sort-ascending(front[1].f_1, ..., front[k].f_1)
values-f_2 ← sort-ascending(front[1].f_2, ..., front[k].f_2)
values-f_1[k + 1] ← y_1^ref
values-f_2[k + 1] ← y_2^ref
/* Part 2: Compute lowest and second lowest height values for each cube
   (with as base area: square ij) */
for all i, j ∈ {1, ..., k} do
   height-lowest[i][j] ← y_3^ref
   height-second-low[i][j] ← y_3^ref
end for
for all i, j, t ∈ {1, ..., k} do
   /* If individual front[t] dominates square ij */
   if (front[t].f_1 ≤ values-f_1[i]) and (front[t].f_2 ≤ values-f_2[j]) then
      /* Update lowest and second lowest height for square ij */
      if (front[t].f_3 < height-lowest[i][j]) then
         height-second-low[i][j] ← height-lowest[i][j]
         height-lowest[i][j] ← front[t].f_3
      else if (front[t].f_3 < height-second-low[i][j]) and
               (front[t].f_3 ≠ height-lowest[i][j]) then
         height-second-low[i][j] ← front[t].f_3
      end if
   end if
end for
/* Part 3: Calculate total S or Sum up partial volume loss due to the removal
   of individual front[t] */
for all t ∈ {1, ..., k} do
   Δ_S[t] ← 0
end for
if (total = true) then
   for all i, j ∈ {1, ..., k} do
      Δ_S[0] ← Δ_S[0] + (values-f_1[i + 1] − values-f_1[i]) · (values-f_2[j + 1] −
      values-f_2[j]) · (y_3^ref − height-lowest[i][j])
   end for
else
   for all i, j ∈ {1, ..., k} do
      owner-number ← 0; owner-index ← −1
      for all t ∈ {1, ..., k} do
         if (front[t].f_1 ≤ values-f_1[i]) and (front[t].f_2 ≤ values-f_2[j]) and
            (front[t].f_3 = height-lowest[i][j]) then
            /* Height of square ij determined by individual front[t] */
            owner-number ← owner-number + 1; owner-index ← t
         end if
      end for
      if (owner-number = 1) then
         Δ_S[owner − index] ← Δ_S[owner − index] + (values-f_1[i + 1] −
         values-f_1[i]) · (values-f_2[j + 1] − values-f_2[j]) · (height-second-low[i][j] −
         height-lowest[i][j])
      end if
   end for
end if                     /* For total S: Δ_S[0], for Δ_S: argmin Δ_S[t] */
```