*Sādhanā* (2020)45:21
https://doi.org/10.1007/s12046-019-1258-y




# Water distribution system design using multi-objective particle swarm optimisation

MAHESH B PATIL[1,*], M NAVEEN NAIDU[2], A VASAN[2] and MURARI R R VARMA[2]

[1]Department of Electrical Engineering, Indian Institute of Technology Bombay, Mumbai, India
[2]Department of Civil Engineering, BITS Pilani, Hyderabad Campus, Hyderabad, India
e-mail: mbpatil@ee.iitb.ac.in

MS received 15 March 2019; revised 6 October 2019; accepted 5 November 2019

**Abstract.** Application of the multi-objective particle swarm optimisation (MOPSO) algorithm to design of water distribution systems is described. An earlier MOPSO algorithm is augmented with (a) local search, (b) a modified strategy for assigning the leader and (c) a modified mutation scheme. For one of the benchmark problems described in the literature, the effect of each of these features on the algorithm performance is demonstrated. The augmented MOPSO algorithm (called MOPSO+) is applied to five benchmark problems, and in each case, non-dominated solutions not reported earlier are found. In addition, for the purpose of comparing Pareto fronts (sets of non-dominated solutions) obtained by different algorithms, a new criterion is suggested, and its usefulness is pointed out with an example. Finally, some suggestions regarding future research directions are made.

**Keywords.** MOPSO; local search; leader selection; water distribution system.

## 1. Introduction

Design of water distribution systems (WDSs) involves various types of challenging optimisation problems, which have been investigated in several research papers to date [1–13]. An important subset of these problems is the two-objective optimisation problem in which pipe diameters are to be determined with the objectives of minimising the total cost and maximising the network resilience. Recently, Wang *et al* [7] reported a systematic collection of twelve benchmark WDS design problems of various complexities to serve as an excellent starting point for researchers to try out new multi-objective optimisation algorithms and compare their performance to the results available in the literature. Furthermore, they have made the corresponding EPANET [14] files and the best-known Pareto front (PF) data available in the public domain [15].

Of the twelve benchmark problems presented in [7], six are summarised in table 1. For a "small problem" (SP), the size of the search space is relatively small, e.g., for the TLN problem, it is $14^8 = 1.48 \times 10^9$, and it is possible to compute the cost and resilience for each of the $14^8$ possible networks to obtain the true PF. In larger problems, including medium and intermediate problems (marked as MP and IP, respectively, in the table), it is not possible to compute the objective function values for each possible network configuration, and a multi-objective evolutionary algorithm (MOEA) is employed to obtain the approximate PF (i.e., set of non-dominated (ND) solutions). In these cases, the true PF is not known, but following [7], we will often use the term "Pareto front" (PF) to refer to the set of ND solutions obtained by the concerned algorithm.

Five MOEAs were used in [7] for each of the benchmark problems. The ND solutions obtained by the five algorithms were combined, and dominated solutions were removed to form the "best-known" PF. The number of solutions ($N_{PF}$) in the best-known PF thus obtained is listed in table 1 for each benchmark problem. The percentage of solutions that contribute to $N_{PF}$ for each problem by each of the five algorithms is given in table 2. It can be observed that, except for TLN and HAN networks, no single algorithm is able to obtain all of the solutions in the best-known PF.

The computation effort involved in obtaining the best-known PF can be described in terms of the total number of function evaluations $N_{FE}^{net}$ by all five algorithms together, since no single algorithm is found to be adequate in general. For the problems in the MP category, $N_{FE}$ was restricted to 600,000 for each of the five algorithms [7], and 30 independent (randomly initiated) runs were performed for each benchmark problem. The total $N_{FE}$ is therefore $N_{FE}^{net} = 600{,}000 \times 30 \times 5 = 90$ million (90 M), as shown in table 1.

The purpose of this paper is to present a modified multi-objective particle swarm optimisation algorithm (MOPSO+) for WDS design. In particular, we present

*For correspondence
Published online: 22 January 2020

**Table 1.** Summary of some of the benchmark WDS design problems presented in [7].

| Network | Type | $N_{pipes}$ | $N_{dia}$ | Search space size | $N_{FE}^{net}$ | $N_{PF}$ |
|---|---|---|---|---|---|---|
| TLN (Two-loop network) | SP | 8 | 14 | $1.48 \times 10^{9}$ | 15M | 128 |
| NYT (New York tunnel network) | MP | 21 | 16 | $1.93 \times 10^{25}$ | 90M | 627 |
| BLA (Blacksburg network) | MP | 23 | 14 | $2.30 \times 10^{26}$ | 90M | 901 |
| HAN (Hanoi network) | MP | 34 | 6 | $2.87 \times 10^{26}$ | 90M | 575 |
| GOY (GoYang network) | MP | 30 | 8 | $1.24 \times 10^{27}$ | 90M | 480 |
| PES (Pescara network) | IP | 99 | 13 | $1.91 \times 10^{110}$ | 150M | 782 |

**Table 2.** Percentage of solutions contributed by five MOEAs to the best-known Pareto front [7].

| | Contribution (%) | | | | |
|---|---|---|---|---|---|
| Network | NSGA-II | $\epsilon$-MOEA | $\epsilon$-NSGA-II | AMALGAM | Borg |
| TLN (SP) | 100.0 | 83.1 | 83.1 | 98.7 | 84.4 |
| NYT (MP) | 93.8 | 17.9 | 24.8 | 91.7 | 20.7 |
| BLA (MP) | 77.0 | 30.3 | 26.3 | 90.1 | 28.3 |
| HAN (MP) | 100.0 | 20.5 | 23.1 | 89.7 | 25.6 |
| GOY (MP) | 43.3 | 3.0 | 58.2 | 85.1 | 22.4 |
| PES (IP) | 38.1 | 27.0 | 10.7 | 18.6 | 23.3 |

results for some of the benchmark problems described in [7] and show that the proposed algorithm gives better solutions than the best-known PFs of [7]. The paper is organised as follows. In section 2, we present a new scheme for comparing PFs produced by two algorithms, to be used in subsequent sections to gauge the performance of the MOPSO+ algorithm. In section 3, we describe the local search (LS) scheme implemented in MOPSO+. In section 4, we describe the overall MOPSO+ algorithm and point out specifically the changes with respect to the original MOPSO algorithm presented in [16]. We compare the results obtained using MOPSO+ to the best-known PFs for MPs in section 5. We then point out in section 6 limitations of the LS scheme for larger problems, propose a suitable modification and show its effectiveness for one of the benchmark problems in the IP category [7]. Finally, we present conclusion of this work in section 7 and comment on related future work.

## 2. Comparison of PFs

In comparing the performance of an MOEA to another with respect to a given benchmark problem, we are mainly interested in (a) computation time and (b) the "quality" of the set of ND solutions. Computation time depends on several factors such as the hardware used, programming environment (compiled or interpreted) and whether parallelisation is employed. Because these factors vary, a more objective metric, viz., the total number of function evaluations $N_{FE}^{net}$ is used in [7] and also in this work.

To judge the quality of the ND set obtained by an algorithm, the following metrics are commonly used [16, 17] when the decision variables are continuous: (a) generational distance, a measure of how far the obtained ND solutions are from the true PF, (b) spacing, a measure of the spread of solutions throughout the ND set and (c) error ratio, the percentage of the ND solutions that do not belong to the true Pareto-optimal set.

In the context of the benchmark WDS problems considered here, the true PF is not known. In any case, these metrics are of limited value for the WDS problem because of the discrete nature of the decision variables. From a practical perspective, apart from cost and resilience, the decision maker may need to take into account other considerations for implementation [4], and it is important to make a large number of options available to the decision maker. A more meaningful metric is therefore the number of solutions in the best-known PF, which the concerned algorithm can contribute [7]. Based on this idea, we propose some new metrics for comparing the PFs obtained by two MOEAs for a given benchmark problem.

Consider a two-objective problem in which the objective functions $f_1$ and $f_2$ are to be maximised and minimised, respectively. Figure 1 shows an example with two sets of ND solutions, referred to as PF-A and PF-B. The combined set of ND solutions (marked as "combined PF" in the figure) is obtained by combining solutions from PF-A and PF-B and then removing solutions dominated by any other solutions. The combined PF contains 14 solutions. PF-A, which contains a total of 10 solutions ($A_1$–$A_{10}$), contributes 5 unique solutions ($A_2$, $A_3$, $A_6$, $A_9$, $A_{10}$), i.e., solutions that

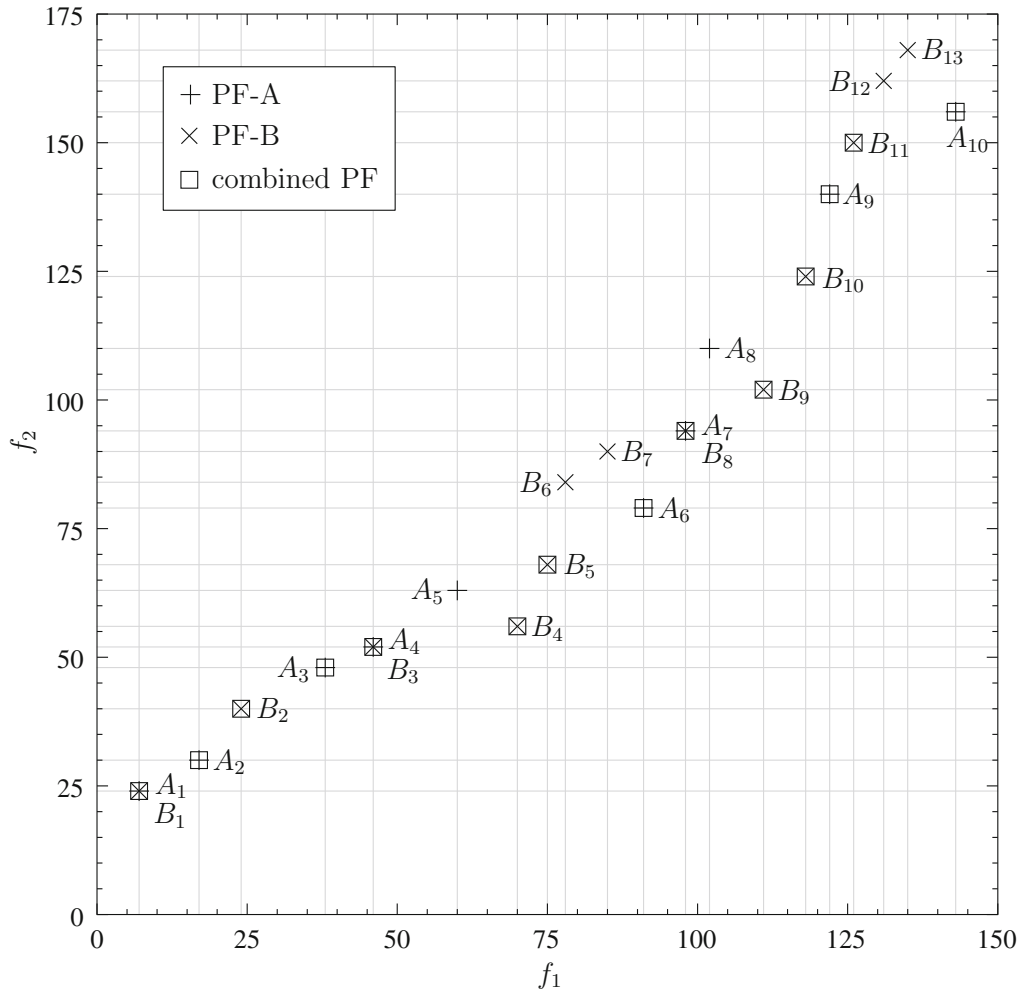


**Figure 1.** Example showing two sets of ND solutions (Pareto fronts) A and B.

are not present in PF-B. Similarly, PF-B, which contains a total of 13 solutions ($B_1$–$B_{13}$), contributes 6 unique solutions ($B_2$, $B_4$, $B_5$, $B_9$, $B_{10}$ $B_{11}$) to the combined PF. The number of common solutions in the combined PF, i.e., solutions contributed by PF-A as well as PF-B, is 3 ($A_1/B_1$, $A_4/B_3$, $A_7/B_8$). Note that some of the solutions have been rejected from the original PFs, viz., 2 solutions ($A_5$, $A_8$) from PF-A and 4 solutions ($B_6$, $B_7$, $B_{12}$ $B_{13}$) from PF-B.

We use the following nomenclature to describe this scenario: $N_{\mathrm{A}}^{\mathrm{t}} = 10$, $N_{\mathrm{A}}^{\mathrm{u}} = 5$, $N_{\mathrm{A}}^{\mathrm{c}} = 3$, $N_{\mathrm{A}}^{\mathrm{r}} = 2$ for PF-A, and $N_{\mathrm{B}}^{\mathrm{t}} = 13$, $N_{\mathrm{B}}^{\mathrm{u}} = 6$, $N_{\mathrm{B}}^{\mathrm{c}} = 3$, $N_{\mathrm{B}}^{\mathrm{r}} = 4$ for PF-B. The superscripts t, u, c, r stand for total, unique, common and rejected, respectively. Note that $N_{\mathrm{A}}^{\mathrm{c}}$ and $N_{\mathrm{B}}^{\mathrm{c}}$ must be equal by definition, and we will denote them by $N^{\mathrm{c}}$. Clearly, for this example, we cannot say that PF-A is better than PF-B (or *vice versa*) since each of them has contributed some unique solutions to the combined set.

Consider now the case $N_{\mathrm{A}}^{\mathrm{u}} = 0$, $N_{\mathrm{B}}^{\mathrm{u}} = 4$. In this case, PF-B is certainly better than PF-A because it contributes some unique solutions while PF-A contributes none. In other words, we can do without the information provided by PF-A since PF-B provides all of the solutions that PF-A would have contributed to the combined front ($N^{\mathrm{c}}$ of them), and $N_{\mathrm{B}}^{\mathrm{u}}$ additional solutions. We will use this concept to compare the PFs obtained by the proposed MOPSO+ algorithm to the best-known PFs [15].

## 3. LS implementation

Memetic algorithms (MA), which combine the exploration capability of an evolutionary algorithm with the exploitation capability of an LS method, have been extensively used for multi-objective optimisation [18–27]. Barlow and Tanyimboh [8] have presented a MA for WDS optimisation. In their work, the genetic algorithm (GA) was used along with local improvement. The child population after every $N_{\mathrm{G}}$ GA iterations was obtained with local and cultural improvement operators using the following procedure.

(a) Choose a subset S of the current ND set.
(b) Select one individual from S for local improvement. Define a scalar fitness function with linear weighting, where the weights are obtained using an estimate of the gradient of the PF. For the selected individual:
    (i) Find the Hooke–Jeeves pattern search direction using this scalar fitness function.
    (ii) Perform the cultural learning step by applying the same pattern search direction to a group of individuals in the current ND set.
(c) Repeat (b) until a sufficient number of children are created.

The authors demonstrated that the introduction of local improvement led to improved convergence speed and solutions better than previously reported.

In this work, we use a simpler LS scheme based on the observations that (a) new ND solutions can be found in the neighbourhood of known ND solutions [22, 27] and (b) LS is effective in exploring the least-crowded areas of the objective space (where a smaller number of ND solutions have been found so far) [21]. In the following, we illustrate the LS scheme used in this work.

Consider the two-variable, two-objective test problem [28] in which

$$\begin{aligned} f_1(\mathbf{x}) &= -x_1^2 + x_2 \\ f_2(\mathbf{x}) &= \frac{1}{2}x_1 + x_2 + 1 \end{aligned} \tag{1}$$

are to be maximised, subject to the constraints specified in [28].

Suppose the current ND set contains three points $A$, $B$, $C$ in figure 2(a). The corresponding objective function values are shown by $P_A$, $P_B$, $P_C$ in figure 2(b). Our goal is to improve the ND set using LS. To this end, we generate four neighbouring points centred around each existing solution, as shown in figure 2(a). The $x_1$ and $x_2$ positions of the neighbours are given by

$$x_1^{\text{new}} = x_1^{\text{old}} \pm \Delta x_1, x_2^{\text{new}} = x_2^{\text{old}} \pm \Delta x_2. \tag{2}$$

The corresponding neighbours in the objective space are shown in figure 2(b). We now find the ND solutions from the entire set consisting of the old solutions and the newly generated solutions. The new ND set is made up of $P_1$, $P_2$, $P_3$, $P_4$ in figure 2(b). Note that the ND set has improved in two ways: (a) the number of solutions has increased from 3 to 4 and (b) the ND points in the new set dominate the old ND points – in this example, all of the old ND points ($P_A$, $P_B$, $P_C$).

It is important to evaluate the neighbours of each solution in the ND set first and then update the ND set by removing dominated solutions [25]. For example, in figure 2(a), suppose we first evaluate the neighbours of $A$ and obtain two new ND points $P_3$ and $P_4$. If the ND set is updated at this stage, solution $B$ will get removed (since $P_B$ is dominated by $P_3$). As a consequence, the neighbours of $B$ will not get evaluated, and the ND solution $P_2$ will not be obtained. To avoid this situation in MOPSO+, we first prepare a list of all solutions whose neighbours are to be explored. Subsequently, we treat the members of this list one by one until the entire list is covered. The new solutions are then used to update the ND set.

For the WDS problems we consider here, the decision variables are the pipe diameters. Each decision variable can take on values 1, 2, ..., $N_{\text{dia}}$ (where $N_{\text{dia}}$ is the number of possible diameters, see table 1), with 1 corresponding to the smallest diameter, 2 to the next larger diameter, etc. For the BLA network, for example, there are 23 decision variables, each taking a value between 1 and 14. We consider two solutions to be neighbours if they differ by $\pm 1$ in one of the decision variables. In the BLA case, therefore, a general

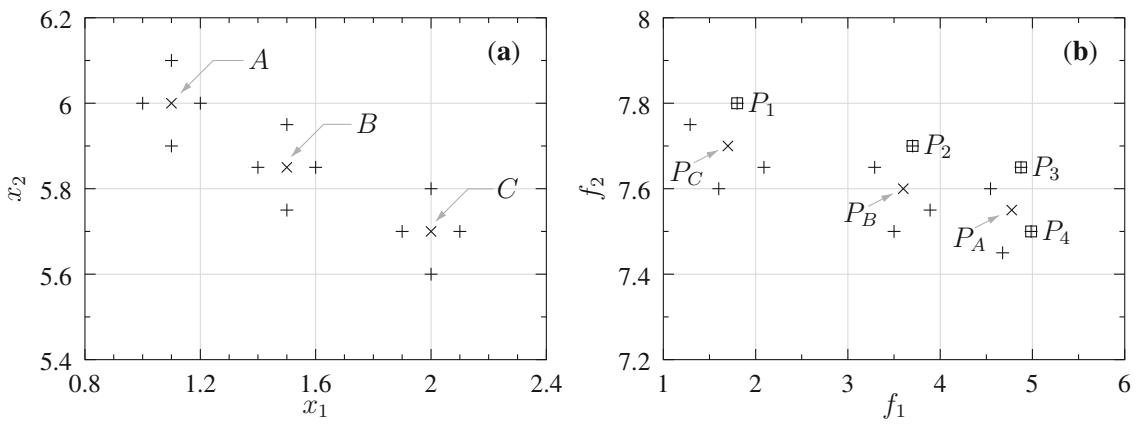


**Figure 2.** Illustration of local search for the optimisation problem given by Eq. (1). (a) Points in decision space and (b) corresponding points in objective space. The original ND points, the points generated by local search and the new ND points are marked with crosses, pluses and squares, respectively.

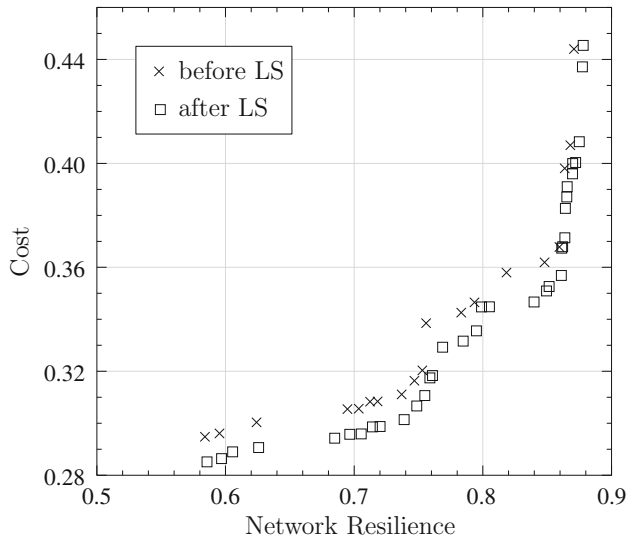


**Figure 3.** ND set for the BLA network at an early stage of the MOEA before and after applying local search (cost is in millions of USD).

solution not lying on a decision space boundary would have $23 \times 2 = 46$ neighbours. If there are $N'$ solutions in the current ND set, we need to perform $46N'$ function evaluations, each involving a call of the EPANET program for computation of cost and resilience. The results before and after LS are shown in figure 3 for the BLA network at an early stage of the MOEA. The improvement arising from LS can be clearly observed. Before LS, there are 19 solutions in the ND set. After LS, the number of ND solutions has increased substantially (to 35), and the new set is found to dominate all solutions in the old set.

This procedure is obviously expensive. One way to reduce the number of function evaluations is to keep track of points that have already been evaluated. For these points, it is not necessary to repeat function evaluation. However, comparison of a given point with a list of points (in the decision space) is also expensive if it is compared with each point in the list one by one. To make the comparison more efficient, we employ a tree structure to represent the ND set. An example is shown in figure 4. In this case, there are three decision variables $x_1$, $x_2$, $x_3$. Nodes at level $x_k$ are labeled by the $x_k$ values of the solutions. The path traced in checking if the solution $x_1 = 3, x_2 = 1, x_3 = 2$ is present in the ND set is shown with a thick grey line.

## 4. MOPSO+ algorithm

In this paper, we demonstrate the usefulness of a hybrid algorithm using the MOPSO algorithm along with LS for the WDS design problem. Several variations of the MOPSO algorithm have been reported (see [29, 30] for a review). Here, we use the MOPSO algorithm proposed by Coello *et al* [16] with some modifications.

The velocity update formula in MOPSO is similar to that used in single-objective particle swarm optimisation (PSO), and is given by

$$\begin{aligned}\mathbf{v}_i(t) = W\,\mathbf{v}_i(t-1) + C_1 r_1(\mathbf{x}^i_{\text{pbest}} - \mathbf{x}_i(t)) \\ + C_2 r_2(\mathbf{x}_{\text{leader}} - \mathbf{x}_i(t)),\end{aligned} \tag{3}$$

where $t$ denotes the PSO iteration number, $\mathbf{x}_i$ and $\mathbf{v}_i$ are the position and velocity of the $i^{\text{th}}$ particle, respectively, and $r_1$ and $r_2$ are random numbers between 0 and 1. The algorithm parameters are $W$ (inertia weight), $C_1$ (cognitive learning factor) and $C_2$ (social learning factor). In this work, we have used $W = 0.4$, $C_1 = 2$, $C_2 = 2$. It should be pointed out that, although the decision variables in the WSD benchmark problems considered here take on discrete values, they are treated as real numbers in the velocity and position update equations. In computing the particle fitness, each decision variable is converted to the nearest integer (which gives the pipe diameter index for the concerned pipe).

The MOPSO algorithm differs from the single-objective PSO algorithm in the computation of $\mathbf{x}^i_{\text{pbest}}$ (the personal best position of the particle so far) and $\mathbf{x}_{\text{leader}}$ (the position of the leader). In [16], $\mathbf{x}^i_{\text{pbest}}$ is updated in every iteration by comparing the particle's current position to the previous value of $\mathbf{x}^i_{\text{pbest}}$. If the current position dominates, it replaces $\mathbf{x}^i_{\text{pbest}}$. If it is ND with respect to $\mathbf{x}^i_{\text{pbest}}$, then one of them is selected randomly as the next $\mathbf{x}^i_{\text{pbest}}$.

In MOPSO [16], assignment of $\mathbf{x}_{\text{leader}}$ is made using the ND set stored in an external archive (repository). The

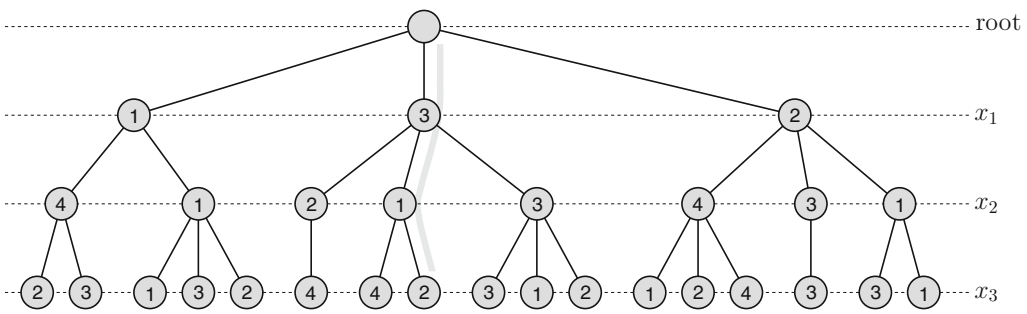


**Figure 4.** Tree structure used for storing solutions in the non-dominated set.

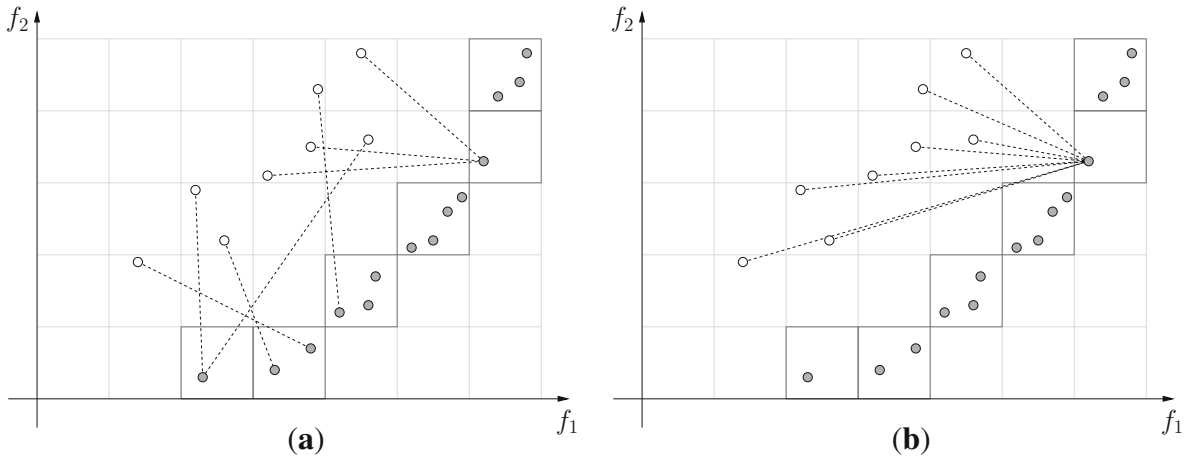


**Figure 5.** Illustration of leader selection procedure: (a) MOPSO algorithm [16] and (b) MOPSO+ algorithm (this work). Hollow circles: PSO particles, filled circles: current ND solutions.

objective space is divided into hypercubes, and each ND solution, depending on its position in the objective space, is assigned one of these hypercubes. For each particle, in each PSO iteration, a leader is selected from the archive, giving preference to ND solutions that occupy less-crowded hypercubes. Roulette-wheel selection is used to first select a hypercube, and one of the ND solutions in that hypercube is picked randomly as the leader. This procedure helps ensure that the ND solutions are well distributed in the objective space.

In addition, a mutation operator is used in [16] to enhance exploration of the search space in the beginning of the search. The mutation rate is made zero as the algorithm converges.

With this background, we now describe modifications made in the proposed MOPSO+ algorithm.

(a) Archive manipulation: The hypergrid approach used in [16] was modified in [31] to avoid changing of grid boundaries and for more efficient use of memory. This new approach, which uses a hypergrid with a fixed cell size and does not involve grid boundaries, is used in the MOPSO+ program.

(b) Leader selection: The leader selection process in MOPSO [16] is illustrated in figure 5(a). In each PSO iteration, each particle is assigned one of the ND particles in the archive, preferring less crowded hypercubes. In MOPSO+, we continue to use the Roulette-wheel selection procedure of the MOPSO algorithm. However, to intensify exploration of the less-crowded regions of the archive, we assign the same leader to all particles, as shown in figure 5(b), and keep the same leader for $N_{\text{leader}}^{\text{const}}$ iterations.

(c) Mutation: In PSO, when the velocity and position update steps fail to generate new ND solutions, mutation can be useful [30]. In the context of the WDS benchmark problems, we have observed that there is an initial phase of MOPSO in which the ND set is improved relatively rapidly. However, beyond a certain point, the rate of generation of new solutions drops significantly. For this reason, different mutation schemes have been implemented in MOPSO+ (see figure 6).

In the "constant" option, the mutation probability remains constant (a low value such as 0.01). In the "pulse" option, the probability is made non-zero only for $N_{\text{mut}}$ iterations in the early stages and zero otherwise. In the "periodic" option the probability is made non-zero for $N_{\text{mut}}$ iterations in every $N_{\text{period}}$ iterations, thus periodically encouraging enhanced exploration. The mutation process itself is common in

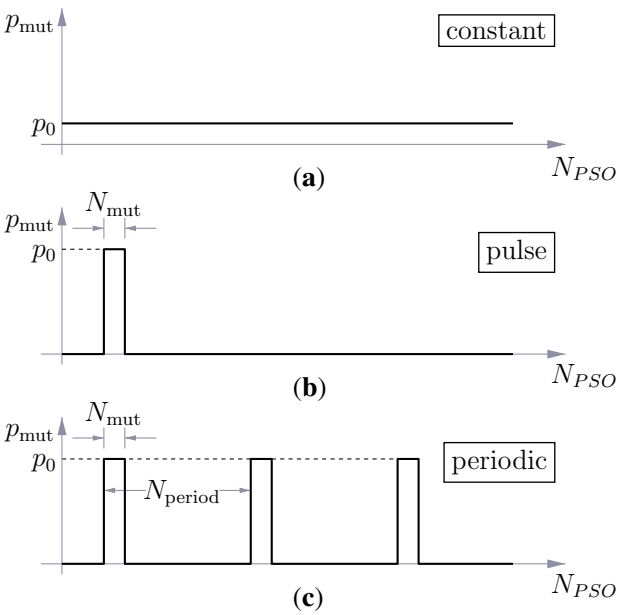


**Figure 6.** Mutation probability versus PSO iteration number for different mutation schemes implemented in MOPSO+.

**Table 3.** Comparison of UExeter and MOPSO+ PFs for the HAN network for different MOPSO+ options.

| | | UExeter (PF-A) | | | | | MOPSO+ (PF-B) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MOPSO+ options | $N_{PSO}^{max}$ | $N_A^t$ | $N_A^a$ | $N_A^u$ | $N_A^r$ | $N_{FE}^{net}$ | $N_B^t$ | $N_B^a$ | $N_B^u$ | $N_B^r$ | $N_{FE}^{net}$ | $N^c$ |
| LS/Leader$^{new}$/M$^{periodic}$ | 10,000 | 575 | 534 | 1 | 41 | 90M | 750 | 748 | 215 | 2 | 74.6M | 533 |
| LS/Leader$^{old}$/M$^{periodic}$ | 10,000 | 575 | 537 | 4 | 38 | 90M | 726 | 714 | 181 | 12 | 71.3M | 533 |
| LS/Leader$^{new}$/M$^{none}$ | 10,000 | 575 | 552 | 88 | 23 | 90M | 629 | 581 | 117 | 48 | 59.6M | 464 |
| LS/Leader$^{new}$/M$^{constant}$ | 10,000 | 575 | 536 | 13 | 39 | 90M | 721 | 703 | 180 | 18 | 69.1M | 523 |
| LS/Leader$^{new}$/M$^{pulse}$ | 10,000 | 575 | 535 | 13 | 40 | 90M | 748 | 729 | 207 | 19 | 71.0M | 522 |
| LS$^{none}$/Leader$^{new}$/M$^{periodic}$ | 10,000 | 575 | 558 | 187 | 17 | 90M | 561 | 452 | 81 | 109 | 40.0M | 371 |
| LS$^{none}$/Leader$^{new}$/M$^{periodic}$ | 20,000 | 575 | 553 | 92 | 22 | 90M | 660 | 577 | 116 | 83 | 80.0M | 461 |

the three cases, and involves changing one of the decision variables of the particle randomly.

(d) LS: The LS operation has been described in section 3. We will refer to that procedure as a "unit local search" (ULS) step. In MOPSO+, LS is implemented as follows:

(i) As seen in section 3, a ULS step can lead to some improvement in the ND set. If it is applied again on the new ND set, further improvement is possible [25]. For this purpose, MOPSO+ allows the ULS to be repeated $N_{LS}^{max}$ times at a given PSO iteration. If, after some ULS steps, it is found that no further generation of new ND solutions is taking place, the LS step is discontinued.

(ii) As explained in section 3, LS is expensive, and it is not practical to perform it in every PSO iteration. In MOPSO+, therefore, LS is performed periodically instead of every iteration. Furthermore, it was observed in the context of the benchmark WDS problems that LS is more effective in the early stages. Based on this observation, a two-stage LS strategy is implemented. From iteration $N_{PSO}^{(1)}$ to $N_{PSO}^{(2)}$, LS is performed every $T_1$ iterations, and after $N_{PSO}^{(2)}$, it is performed every $T_2$ iterations.

## 5. Results for MPs

In this section, we present results obtained with the MOPSO+ algorithm for the "medium" problems (MP) listed in table 1. We first consider the HAN network to understand the effect of the various algorithm parameters. We then choose the set of parameters that gives the best performance for the HAN network and use it for the other three benchmark problems in the MP category, viz., the NYT, BLA and GOY networks. We will refer to the ND sets reported in [16] as "UExeter PFs" and the ND sets given by MOPSO+ as the "MOPSO+ PFs."

Table 3 shows the results (i.e., $N_A^t$, $N_A^u$, etc. as defined in section 2) for the HAN network with $N_p = 200$ (number of particles). For the first six rows of the table, the number of PSO iterations $N_r = 10{,}000$, while for the last row it is 20,000.

The MOPSO+ options mentioned in the first column have the following meaning.

(a) LS: Local search is performed with a period of 100 between $N_{PSO} = 1{,}000$ and 5,000, and with a period of 1,000 thereafter. The parameter $N_{LS}^{max}$ (see section 3) is set to 50.

(b) Leader$^{new}$: The new leader assignment scheme (figure 5(b)) is used with $N_{leader}^{const} = 10$.

(c) Leader$^{old}$: The leader assignment scheme of [16] (figure 5(a)) is used.

(d) M$^{none}$: No mutation is performed.

(e) M$^{constant}$: Constant mutation (figure 6(a)) is performed with $p_0 = 0.02$.

(f) M$^{pulse}$: Pulse mutation (figure 6(b)) is performed with $p_0 = 1$, $N_{mut} = 20$, starting with $N_{PSO} = 1{,}000$.

(g) M$^{periodic}$: Periodic mutation (figure 6(c)) is performed with $p_0 = 1$, $N_{mut} = 20$, $N_{mut}^{period} = 1{,}000$, starting with $N_{PSO} = 1{,}000$.

(h) LS$^{none}$: Local search is not used.

In each case, 20 independent runs of MOPSO+ were performed and the total number of function evaluations (using the EPANET program) $N_{FE}^{net}$ over all independent runs was recorded. We can make the following observations from table 3.

(a) For each set of MOPSO+ options, a substantial number of new ND solutions (given by $N_B^u$) are found.

(b) Comparing the first two rows, we find that the new leader assignment scheme used in MOPSO+ gives a larger $N_B^u$. It also results in a smaller $N_A^u$, which is desirable, as explained in section 3.

(c) Among the different mutation schemes (see rows 1, 3, 4, 5), the periodic scheme gives the best PF.

(d) LS plays a very important role (see rows 1 and 6) in MOPSO+. Without LS, we see that a large number of

UExeter solutions ($N_{\mathrm{A}}^{\mathrm{u}} = 187$) are missed out by MOPSO+. Also, there is a substantial reduction (from 215 to 81) in the number of new solutions found by MOPSO+ when LS is not used.

(e) Although LS has improved the ND set, it has taken a larger number of function evaluations ($N_{\mathrm{FE}}^{\mathrm{net}} = 74.6$M with LS as against $N_{\mathrm{FE}}^{\mathrm{net}} = 40$M without LS, see rows 1 and 6). To make a fair comparison, we have run MOPSO+ without LS with a larger $N_{\mathrm{FE}}^{\mathrm{net}} = 80$M by doubling $N_{\mathrm{PSO}}^{\mathrm{max}}$, the number of PSO iterations. In this situation (row 7), $N_{\mathrm{A}}^{\mathrm{u}}$ and $N_{\mathrm{B}}^{\mathrm{u}}$ are worse than row 1, which implies that improvement due to LS is not simply because of increased $N_{\mathrm{FE}}^{\mathrm{net}}$.

It is instructive to look at the LS process for one specific run for the first row of table 3. At a given PSO iteration at an early stage ($N_{\mathrm{PSO}} = 1{,}100$), the number of accepted solutions ($N_{\mathrm{a}}^{\mathrm{LS}}$), the number of rejected solutions ($N_{\mathrm{r}}^{\mathrm{LS}}$) and the number of solutions in the updated ND set ($N_{\mathrm{PF}}$) are plotted versus $N_{\mathrm{LS}}$, the LS iteration number, in figure 7(a)–(c). At the first LS iteration ($N_{\mathrm{LS}} = 0$), the number of solutions in the ND set is about 550, which means that there are $550 \times 2 \times 34$ (where 34 is the number of decision variables), i.e. 37,400, neighbours if all of the 550 points are interior. Some of these points would lie on the boundaries of the decision space, and they would have less than $2 \times 34$ neighbours each. Nevertheless, it is clear that the number of solutions to be evaluated at the first LS iteration is rather large. Of these, only 138 solutions (figure 7(a)) get accepted in the ND set. About 25,000 (figure 7(b)) are found to be dominated by other solutions in the ND set, and are therefore rejected.

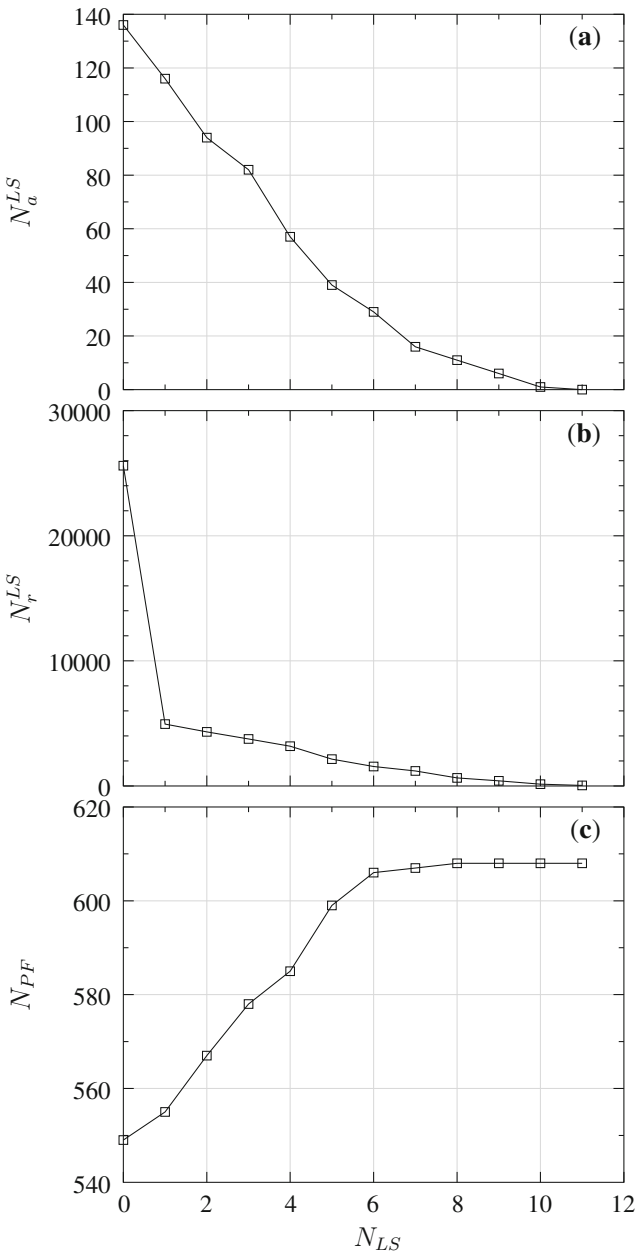


**Figure 7.** Illustration of the local search process at a specific PSO iteration ($N_{\mathrm{PSO}} = 1{,}100$) for the HAN network with algorithm parameters same as those for row 1 of table 3. (a) Number of accepted solutions, (b) number of rejected solutions and (c) number of solutions in the ND set.

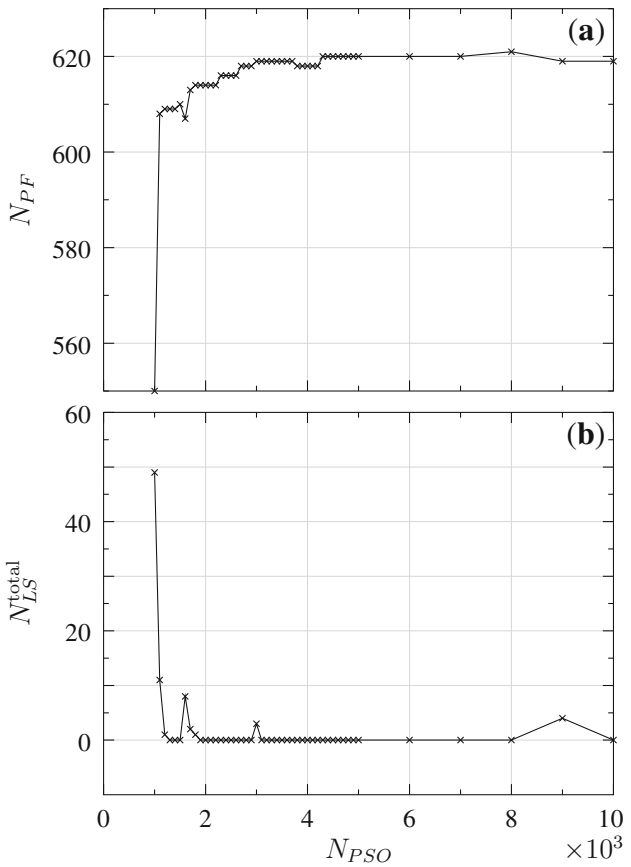


**Figure 8.** (a) Number of ND solutions ($N_{\mathrm{PF}}$) and (b) number of total LS iterations ($N_{\mathrm{LS}}^{\mathrm{total}}$) versus PSO iteration number for the HAN network with algorithm parameters same as those for row 1 of table 3.

**Table 4.** Comparison of UExeter and MOPSO+ PFs for different values of $N_p$ (number of particles), $N_{PSO}^{max}$ (number of PSO iterations) and $N_r$ (number of independent runs) for the NYT network.

| MOPSO+ options | | | UExeter (PF-A) | | | | | MOPSO+ (PF-B) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $N_p$ | $N_{PSO}^{max}$ | $N_r$ | $N_A^t$ | $N_A^a$ | $N_A^u$ | $N_A^r$ | $N_{FE}^{net}$ | $N_B^t$ | $N_B^a$ | $N_B^u$ | $N_B^r$ | $N_{FE}^{net}$ | $N^c$ |
| 200 | 20,000 | 30 | 627 | 599 | 20 | 28 | 90M | 648 | 639 | 60 | 9 | 150.7M | 579 |
| 300 | 20,000 | 30 | 627 | 597 | 28 | 30 | 90M | 670 | 630 | 61 | 40 | 211.3M | 569 |
| 100 | 60,000 | 10 | 627 | 596 | 25 | 31 | 90M | 655 | 635 | 64 | 20 | 76.7M | 571 |
| 100 | 60,000 | 17 | 627 | 595 | 4 | 32 | 90M | 661 | 656 | 65 | 5 | 130.3M | 591 |

**Table 5.** Comparison of UExeter and MOPSO+ PFs for the four MP networks described in [7]. The MOPSO+ algorithm parameters are the same as those for row 1 of table 3.

| | MOPSO+ options | | | UExeter (PF-A) | | | | | MOPSO+ (PF-B) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Network | $N_p$ | $N_{PSO}^{max}$ | $N_r$ | $N_A^t$ | $N_A^a$ | $N_A^u$ | $N_A^r$ | $N_{FE}^{net}$ | $N_B^t$ | $N_B^a$ | $N_B^u$ | $N_B^r$ | $N_{FE}^{net}$ | $N^c$ |
| NYT | 100 | 60,000 | 17 | 627 | 595 | 4 | 32 | 90M | 661 | 656 | 65 | 5 | 130.3M | 591 |
| BLA | 200 | 10,000 | 10 | 901 | 849 | 0 | 52 | 90M | 1045 | 1045 | 196 | 0 | 44.1M | 849 |
| HAN | 200 | 10,000 | 20 | 575 | 534 | 1 | 41 | 90M | 750 | 748 | 215 | 2 | 74.6M | 533 |
| GOY | 100 | 20,000 | 10 | 489 | 444 | 3 | 45 | 90M | 571 | 570 | 129 | 1 | 37.9M | 441 |

During the first LS iteration (i.e., the ULS step described earlier), a list of all solutions being evaluated is prepared using the tree format shown in figure 4. In the subsequent LS iterations the neighbours of the updated ND solutions are compared against this list, and only those not found in the list are evaluated. Therefore, a much smaller number of solutions need to be tried in the subsequent LS iterations. For example, for $N_{LS} = 1$, only about 5,000 solutions need to be evaluated. This process continues until either $N_{LS}^{max}$ iterations are completed or when an LS iteration fails to generate any new ND solutions. For the example shown in figure 7, the latter situation is reached (at $N_{LS} = 11$). With each LS iteration the ND set gets improved, and in this specific example the number of solutions in the ND set also increases (figure 7(c)) with each LS iteration.

Figure 8(a) shows the variation of the size of the ND set ($N_{PF}$), and figure 8(b) shows the total number of LS iterations tried ($N_{LS}^{total}$) as the MOPSO+ algorithm proceeds. Note that, as mentioned earlier, the period of LS is low (100) up to $N_{PSO} = 5{,}000$ and high (1,000) thereafter, and that is the reason for the larger density of data points up to $N_{PSO} = 5{,}000$ in figure 8(b). For the first LS step (at $N_{PSO} = 1{,}000$), $N_{LS}^{total} = N_{LS}^{max} = 50$. Beyond this point, generation of ND solutions due to LS slows down and after about 2,000 PSO iterations, we see that $N_{LS}^{total}$ mostly remains zero (which means only one LS iteration was tried, which failed to produce new ND points). This points to a side benefit of using LS: it provides a necessary (but not sufficient) condition for convergence of an MOEA. If LS yields additional ND points, we can say that the MOEA has not converged.

It may be noted that the number of additional function evaluations due to LS depends on how the particles evolve with and without LS. As mentioned earlier, the LS step is computationally intensive. For example, comparing rows 1 and 6 of table 3, we see that $N_{FE}^{net}$ has increased substantially (from 40M to 74.6M) when LS is included. The difference (34.6M) is due to function evaluations involved in the LS steps.

We now use the MOPSO+ algorithm parameters that have been found to work well for the HAN problem (option LS/Leader$^{new}$/M$^{periodic}$ in table 3) for optimisation of the other MPs described in [16], viz., the NYT, BLA and GOY networks. The results for the NYT network are shown in table 4. We observe that the $N_B^u$ values are comparable in the four cases listed in the table. However, for $N_p = 100$, $N_{PSO}^{max} = 60{,}000$ and $N_r = 17$, $N_A^u$ is substantially lower than in the other three cases.

Comparing rows 1 and 2 of the table, we conclude that increasing the number of particles is not necessarily beneficial, which is also pointed out in [7] for other MOEAs. A systematic study on the effect of $N_p$ and $N_{PSO}^{max}$ on the performance of MOPSO+ has not been carried out in this work. Instead, we present the results obtained after a limited experimentation with these parameters.

Table 5 shows the results for the four MP benchmark problems described in [7]. In each case, a substantial number of new ND solutions ($N_B^u$) have been found by MOPSO+. Furthermore, almost all ND solutions in the UExeter PFs are covered by MOPSO+ (see the $N_A^u$ column in the table). The improvements in the PFs have been obtained with a smaller $N_{FE}^{net}$ for three networks and with a larger $N_{FE}^{net}$ for the NYT network. These results, overall, point to the usefulness of the MOPSO+ approach for the WDS design problem.

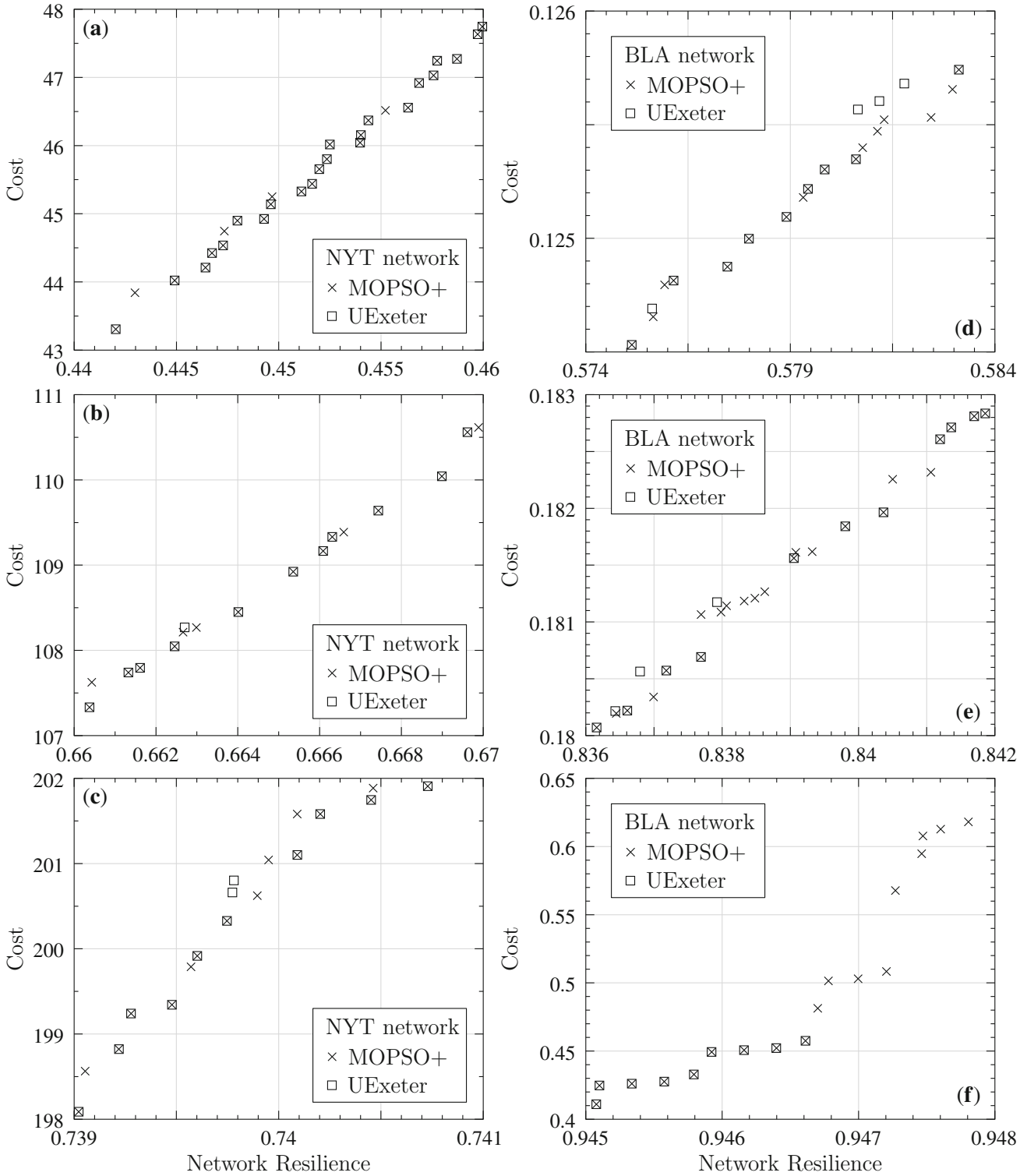


**Figure 9.** UExeter [15] and MOPSO+ ND sets for the NYT and BLA networks for different ranges of network resilience.

Figures 9(a)–(f) and 10(a)–(f) show expanded plots of the UExeter and MOPSO+ PFs. It can be seen that some of the MOPSO+ solutions dominate some of the UExeter solutions.

Even though many of the unique MOPSO+ solutions (i.e., solutions present in the MOPSO+ PFs but not in the UExeter PFs) are close in the objective space to the previously known UExeter solutions, they can differ

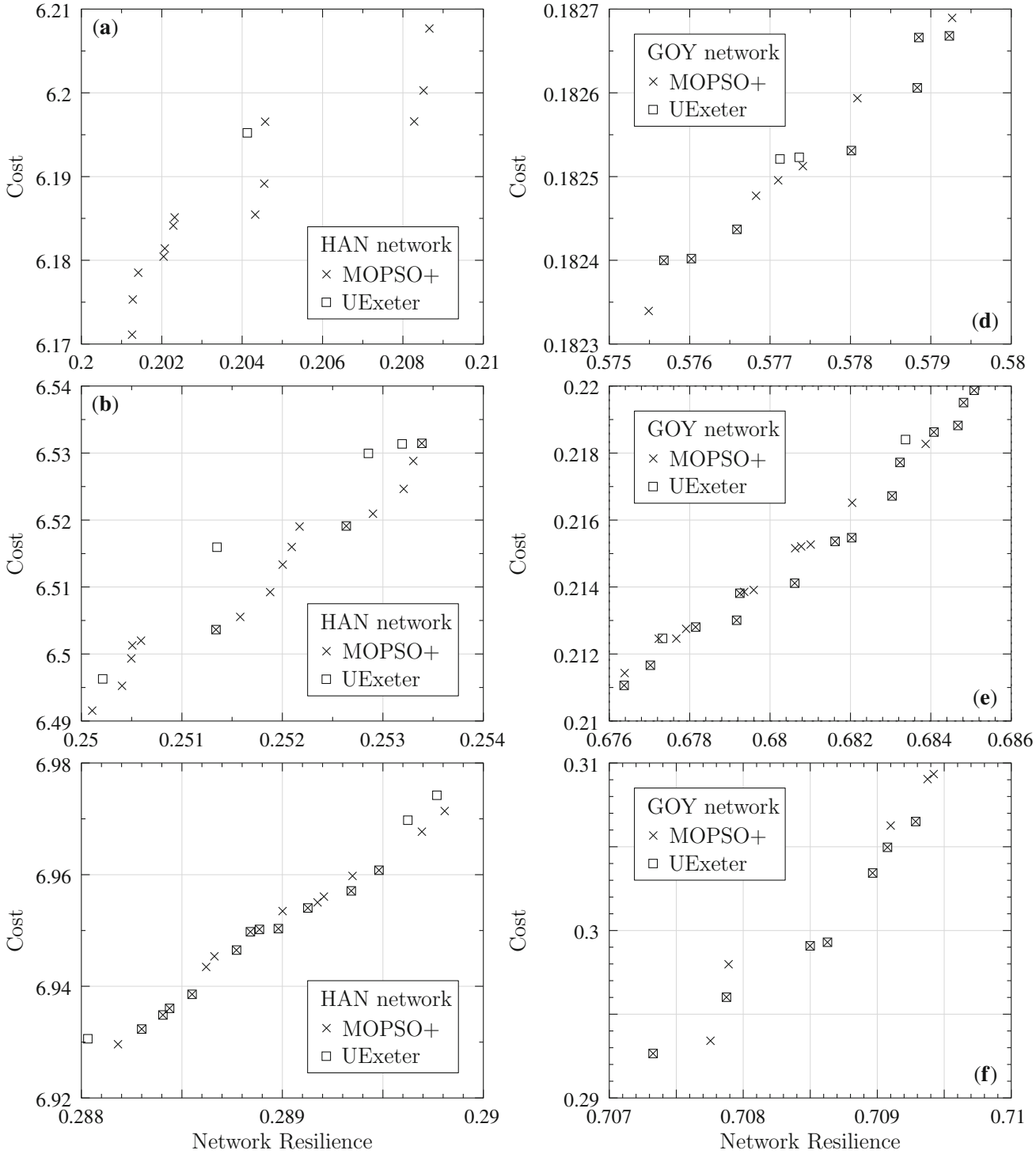


**Figure 10.** UExeter [15] and MOPSO+ ND sets for the HAN and GOY networks for different ranges of network resilience.

substantially in the decision space. For example, some of the ND solutions obtained by MOPSO+ for the BLA network are shown in table 6. Of these, solutions 2, 3, 4 are also present in the UExeter PF while solutions 1, 5, 6, 7 are new. Although the cost and resilience values for solutions 1 and 2 differ only in the fourth decimal place, we observe that these two solutions vary substantially in the decision space – four diameter values, viz., $d_{11}$, $d_{15}$, $d_{18}$ and $d_{23}$, are different. This could have important implications in practice because of non-technical issues such as land acquisition, and a wider choice of ND solutions may well open up previously unknown options

**Table 6.** Objective function and decision variable values (diameters in mm) for a few solutions of figure 9(d) obtained by MOPSO+ for the BLA network

| | sol. 1 | sol. 2 | sol. 3 | sol. 4 | sol. 5 | sol. 6 | sol. 7 |
|---|---|---|---|---|---|---|---|
| Resilience | 0.57931 | 0.57943 | 0.57984 | 0.58060 | 0.58077 | 0.58113 | 0.58129 |
| Cost | 0.12518 | 0.12522 | 0.12530 | 0.12535 | 0.12540 | 0.12547 | 0.12552 |
| $d_1$ | 254.0 | 254.0 | 254.0 | 254.0 | 254.0 | 254.0 | 254.0 |
| $d_2$ | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 |
| $d_3$ | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 |
| $d_4$ | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 | 203.2 |
| $d_5$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_6$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_7$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_8$ | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 |
| $d_9$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_{10}$ | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 | 101.6 |
| $d_{11}$ | 101.6 | 76.2 | 101.6 | 76.2 | 76.2 | 76.2 | 76.2 |
| $d_{12}$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_{13}$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_{14}$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_{15}$ | 76.2 | 101.6 | 101.6 | 76.2 | 76.2 | 101.6 | 101.6 |
| $d_{16}$ | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 |
| $d_{17}$ | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 | 50.8 |
| $d_{18}$ | 76.2 | 50.8 | 76.2 | 25.4 | 76.2 | 25.4 | 76.2 |
| $d_{19}$ | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 |
| $d_{20}$ | 101.6 | 101.6 | 101.6 | 152.4 | 101.6 | 152.4 | 101.6 |
| $d_{21}$ | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 |
| $d_{22}$ | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 | 152.4 |
| $d_{23}$ | 101.6 | 152.4 | 101.6 | 101.6 | 152.4 | 101.6 | 152.4 |

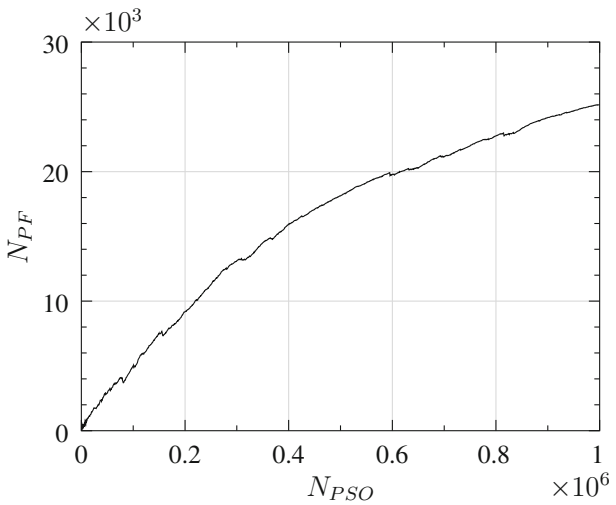


**Figure 11.** Number of ND solutions ($N_{\text{PF}}$) versus PSO iteration number for the PES network.

for the decision maker. The metrics proposed in section 2 therefore seem to be more attractive from a practical perspective, than metrics such as average cost, lowest cost, standard deviation or spread.

## 6. Results for an IP

As seen in section 3, LS is an expensive step, and with a large number of decision variables (in our context, the number of pipes) involved in the networks of IP and LP categories in [7], the repeated ULS procedure described for the MP category (see section 4) can become prohibitively expensive. In this section, we will consider one of the IPs, the PES network (see table 1). Since there are 99 decision variables, LS around one solution involves $99 \times 2 = 198$ function evaluations. As we shall see, the ND set for this network is much larger than the MP networks, making even one ULS step involving evaluation of all neighbours of all solutions in the ND set consume significant computation time.

In view of this difficulty, the ND set is only partially explored during LS (see [22, 27]), and only one ULS step is performed. In particular, $N_{\text{ND}}^{\text{max}}$ solutions from the current ND set are randomly selected and all neighbours of the selected solutions are evaluated. Apart from this change, other algorithm details remain the same as before.

For the PES problem, 10 independent runs of MOPSO+ were performed with $N_{\text{p}} = 200$ and $N_{\text{PSO}} = 1{,}000{,}000$. Periodic mutation was employed, with $p_0 = 1$, $N_{\text{mut}} = 50$ and $N_{\text{mut}}^{\text{period}} = 1{,}000$, starting with $N_{\text{PSO}} = 1{,}000$. The new

**Table 7.** Comparison of UExeter and MOPSO+ PFs for different values of $N_r$ (number of independent runs) for the PES network.

| | UExeter (PF-A) | | | | | MOPSO+ (PF-B) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $N_r$ | $N_A^l$ | $N_A^a$ | $N_A^u$ | $N_A^r$ | $N_{FE}^{met}$ | $N_B^l$ | $N_B^a$ | $N_B^u$ | $N_B^r$ | $N_{FE}^{met}$ | $N^c$ |
| 1 | 782 | 307 | 292 | 475 | 150M | 21,536 | 12,161 | 12,146 | 9,375 | 269M | 15 |
| 2 | 782 | 190 | 175 | 592 | 150M | 20,736 | 14,856 | 14,841 | 5,880 | 538M | 15 |
| 3 | 782 | 167 | 152 | 615 | 150M | 22,949 | 20,081 | 20,066 | 2,868 | 808M | 15 |
| 4 | 782 | 166 | 152 | 616 | 150M | 22,962 | 20,094 | 20,080 | 2,868 | 1,077M | 14 |
| 5 | 782 | 39 | 22 | 743 | 150M | 23,267 | 23,168 | 23,151 | 99 | 1,348M | 17 |
| 6 | 782 | 37 | 19 | 745 | 150M | 24,643 | 24,547 | 24,529 | 96 | 1,619M | 18 |
| 7 | 782 | 36 | 19 | 746 | 150M | 25,607 | 25,510 | 25,493 | 97 | 1,889M | 17 |
| 8 | 782 | 36 | 19 | 746 | 150M | 26,145 | 26,048 | 26,031 | 97 | 2,160M | 17 |
| 9 | 782 | 35 | 18 | 747 | 150M | 26,488 | 26,399 | 26,382 | 89 | 2,431M | 17 |
| 10 | 782 | 35 | 18 | 747 | 150M | 30,267 | 30,178 | 30,161 | 89 | 2,703M | 17 |

scheme for finding the leader (see section 4) is used, with $N_{leader}^{const} = 10$. LS was performed with a period of 1,000 between $N_{PSO} = 1{,}000$ and 10,000, and with a period of 10,000 thereafter. $N_{ND}^{max}$ was set to 200, which means only 200 solutions from the current ND set are randomly picked for any LS step.

Figure 11 shows $N_{PF}$ (the number of ND solutions) versus PSO iteration number for one independent run. A

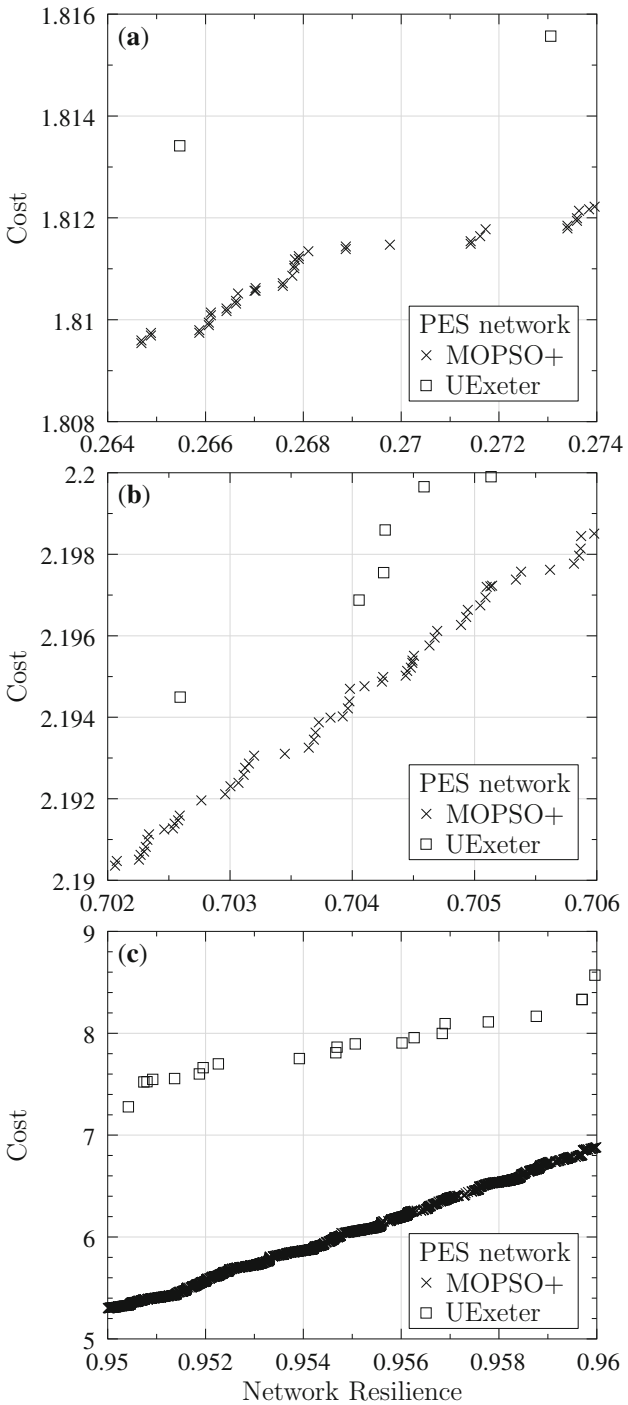


**Figure 12.** UExeter [15] and MOPSO+ ND sets for the PES network for different ranges of network resilience.

much larger $N_{PF}$ (about 25,000) is obtained for this network. Note that, in contrast with figure 8 for the HAN network, $N_{PF}$ has not saturated in this case, indicating that $N_{PF}$ may increase further with additional PSO iterations.

Table 7 shows the effect of $N_r$ (the number of independent runs) on the performance of MOPSO+. With a single run, $N_B^u = 12,146$ new solutions are generated. However, $N_A^u = 292$ solutions from the UExeter PF are missed out by MOPSO+. The number of function evaluations for this run is 269M, out of which $N_p N_r = 200$M are due to the PSO part and 69M are due to LS. This value of $N_{FE}^{net}$ is already larger than the UExeter value of 150M. As more independent runs are performed, $N_{FE}^{net}$ increases proportionately. However, the number of additional ND solutions increases with $N_r$, and the number of UExeter ND solutions ($N_A^u$) missed by MOPSO+ decreases, both factors being favourable. For 10 runs, $N_{FE}^{net}$ for MOPSO+ is about 18 times larger than UExeter. Considering the large number of new ND solutions (about 38 times larger than UExeter), the additional computational effort seems worthwhile.

Because of the longer computation time taken for obtaining the MOPSO+ results, it is not possible to say that the performance of MOPSO+ is superior to that of UExeter for the PES problem. However, the large number of ND solutions generated by MOPSO+ points to its usefulness.

Some of the ND solutions obtained by MOPSO+ are shown in figure 12(a)–(c), along with UExeter solutions. It can be seen that the density of solutions obtained by MOPSO+ is larger. Also, for some values of network resilience (see figure 12(c)), the cost for some of the MOPSO+ solutions is lower by as much as 25%.

## 7. Conclusions

In summary, a hybrid algorithm (MOPSO+) comprising a multi-objective particle swarm optimisation algorithm and local search is presented for water distribution system design. Four medium and one intermediate benchmark networks are considered, and for each of them, a significant number of new ND solutions have been obtained using the proposed algorithm, thus improving the best-known PFs for these problems. For medium networks, the total number of function evaluations for MOPSO+ was of the same order as in [7]. For the intermediate network considered here, although the total $N_{FE}$ is much larger (18 times) for MOPSO+, the large number of new solutions generated by MOPSO+ justifies the additional computational effort.

Based on the results presented here, MOPSO+ seems to be an attractive option for WDS design. Some important issues for future research in this area are listed here:

(a) Performance of MOPSO+ for other networks in the IP and LP category needs to be investigated.
(b) A unified set of algorithmic parameters for all problems of a particular complexity (MP, IP, LP) is desirable. More work is required to explore this possibility.
(c) The LS mechanism introduced here can be applied to other multi-objective optimisation problems with discrete decision variables.
(d) In order to speed up convergence, techniques such as reduction of search space [8] and high-quality initial population [11] could be combined with the MOPSO+ algorithm.
(e) Other promising mutation schemes, such as the "elitist mutation" scheme presented in [2], could be combined with LS for possibly improving the performance further.

## Acknowledgements

This work was partially supported by Council of Scientific and Industrial Research (CSIR) Grant No.: 22(0723)/17/EMR-II, New Delhi, India.

## List of abbreviations

| | |
|---|---|
| WDS | water distribution system |
| SP | small problem |
| MP | medium problem |
| IP | intermediate problem |
| PSO | particle swarm optimisation |
| MOPSO | multi-objective particle swarm optimisation |
| MOPSO+ | MOPSO with additional features |
| MA | memetic algorithm |
| MOEA | multi-objective evolutionary algorithm |
| GA | genetic algorithm |
| ND | non-dominated |
| PF | Pareto front (set of ND solutions obtained by the concerned algorithm) |
| LS | local search |
| ULS | unit local search |

## References

[1] Farmani R, Walters G A and Savic D A 2005 Trade-off between total cost and reliability for anytown water distribution network. *J. Water Res. Plan. Manag.* 131(3): 161–171

[2] Reddy M J and Kumar D N 2007 An efficient multi-objective optimization algorithm based on swarm intelligence for engineering design. *Eng. Optim.* 39(1): 49–68

[3] Kadu M S, Gupta R and Bhave P R 2008 Optimal design of water networks using a modified genetic algorithm with reduction in search space. *J. Water Resour. Plan. Manag.* 134(2): 147–160

[4] Montalvo I, Izquierdo J, Schwarze S and Pérez-García R 2010 Multi-objective particle swarm optimization applied to water distribution systems design: an approach with human interaction. *Math. Comput. Model.* 52(7-8): 1219–1227

[5] Raad D N 2011 Multi-objective optimisation of water distribution systems design using metaheuristics, Ph.D. dissertation, Stellenbosch: University of Stellenbosch

[6] Ahmadi M, Arabi M, Hoag D L and Engel B A 2013 A mixed discrete-continuous variable multiobjective genetic algorithm for targeted implementation of nonpoint source pollution control practices. *Water Resour. Res.* 49(12): 8344–8356

[7] Wang Q, Guidolin M, Savic D and Kapelan Z 2014 Two-objective design of benchmark problems of a water distribution system via MOEAs: Towards the best-known approximation of the true pareto front. *J. Water Resour. Plan. Manag.* 141(3): 04014060

[8] Barlow E and Tanyimboh T T 2014 Multiobjective memetic algorithm applied to the optimisation of water distribution systems. *Water Resour. Manag.* 28(8): 2229–2242

[9] Morley M and Tricarico C 2014 A comparison of population-based optimization techniques for water distribution system expansion and operation. *Procedia Eng.* 89: 13–20

[10] Mora-Melia D, Iglesias-Rey P L, Martinez-Solano F J and Ballesteros-Pérez P 2015 Efficiency of evolutionary algorithms in water network pipe sizing. *Water Resour. Manag.* 29(13): 4817–4831

[11] Bi W, Dandy G C and Maier H R 2016 Use of domain knowledge to increase the convergence rate of evolutionary algorithms for optimizing the cost and resilience of water distribution systems. *J. Water Resour. Plan. Manag.* 142(9): 04016027

[12] Moosavian N and Lence B 2016 Nondominated sorting differential evolution algorithms for multiobjective optimization of water distribution systems. *J. Water Resour. Plan. Manag.* 143(4): 04016082

[13] Zheng F, Zecchin A C, Newman J P, Maier H R and Dandy G C 2017 An adaptive convergence-trajectory controlled ant colony optimization algorithm with application to water distribution system design problems. *IEEE Trans. Evol. Comput.* 21(5): 773–791

[14] Rossman L A *et al* 2000 EPANET 2: Users Manual

[15] Centre for water systems. [Online]. Available: http://emps.exeter.ac.uk/engineering/research/cws/resources/benchmarks/design-resiliance-pareto-fronts/

[16] Coello C A C, Pulido G T and Lechuga M S 2004 Handling multiple objectives with particle swarm optimization. *IEEE Trans. Evol. Comput.* 8(3): 256–279

[17] Deb K 2009 *Multi-objective Optimization Using Evolutionary Algorithms*. Chichester: J. Wiley and Sons, Ltd.

[18] Ishibuchi H and Yoshida T 2002 Hybrid evolutionary multi-objective optimization algorithms. In: *HIS*, pp. 163–172

[19] Knowles J and Corne D 2005 Memetic algorithms for multiobjective optimization: issues, methods and prospects. In: *Recent advances in memetic algorithms*. Springer, pp. 313–352

[20] Sindhya K, Sinha A, Deb K and Miettinen K 2009 Local search based evolutionary multi-objective optimization algorithm for constrained and unconstrained problems. In: *2009 IEEE Congress on Evolutionary Computation*. IEEE, pp. 2919–2926

[21] Mousa A A, El-Shorbagy M A and Abd-El-Wahed W F 2012 Local search based hybrid particle swarm optimization algorithm for multiobjective optimization. *Swarm Evol. Comput.* 3: 1–14

[22] Liefooghe A, Humeau J, Mesmoudi S, Jourdan L and Talbi E 2012 On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems. *J. Heuristics* 18(2): 317–352

[23] Dubois-Lacoste L J, López-Ibánez M and Stützle T 2013 Combining two search paradigms for multi-objective optimization: Two-phase and pareto local search. In: *Hybrid Metaheuristics*. Springer, pp. 97–117

[24] Inja M, Kooijman C, de Waard M, Roijers D M and Whiteson S 2014 Queued pareto local search for multi-objective optimization. In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 589–599

[25] Maler O and Srivastav A 2016 Double archive pareto local search. In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, pp. 1–7

[26] Mansour I B, Alaya I and Tagina M 2017 Chebyshev-based iterated local search for multi-objective optimization. In: *2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*. IEEE, pp. 163–170

[27] Jaszkiewicz A 2018 Many-objective pareto local search. *Eur. J. Oper. Res.* 271(3): 1001–1013

[28] Kita H, Yabumoto Y, Mori N and Nishikawa Y 1996 Multi-objective optimization by means of the thermodynamical genetic algorithm. In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 504–512

[29] Sierra M R and Coello C A C 2005 Improving PSO-based multi-objective optimization using crowding, mutation and $\epsilon$-dominance. In: *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, pp. 505–519

[30] Parsopoulos K E and Vrahatis M N 2008 Multi-objective particles swarm optimization approaches. In: *Multi-objective Optimization in Computational Intelligence: Theory and practice*. IGI global, pp. 20–42

[31] Patil M B Using external archive for improved performance in multi-objective optimization, submitted to Sādhanā