High-Performance Computing for the Optimization of High-Pressure Thermal treatments in Food Industry

M. R. Ferrández · S. Puertas-Martín · J. L. Redondo · B. Ivorra · A. M. Ramos · P. M. Ortigosa

Preprint version in Optimization Online - Published version in J Supercomput (2018): https://doi.org/10.1007/s11227-018-2351-4

Abstract In Food Industry, the combined treatments based on high-pressure and temperature (HPT) are frequently used to increment the durability of the products without damaging their good properties. However, achieving a reasonable compromise between conservation and quality is usually a challenging task. In a previous work, we proposed a decision tool which solves a multi-objective optimization problem providing a set of good configurations for the HPT equipment in order to adapt the final product to different quality scenarios. The considered optimizer is a population-based evolutionary algorithm that takes the decision maker preferences into account. Nevertheless, when the number of solutions demanded by the decision maker is very large or a high precision is required, the computational time needed by such an algorithm may not be negligible at all. In this work, a parallel version of the optimizer has been designed. This parallel algorithm allows obtaining a greater number of solutions by working with more individuals in the population in reasonable computing times. Additionally, using it, we can consider more iterations in the optimization process that lead us to better distributed and more accurate solutions. A computational experiment shows the good performance of the proposed method.

Keywords High-Performance Computing · Multi-Objective Optimization · Genetic Algorithms · High-Pressure · Food Industry

M. R. Ferrández · S. Puertas-Martín · J. L. Redondo · P. M. Ortigosa
Dept. de Informática, Universidad de Almería, ceiA3, Ctra. Sacramento, La Cañada de San Urbano, 04120 Almería, Spain
E-mail: mrferrandez@ual.es, savinspm@ual.es, jlredondo@ual.es, ortigosa@ual.es

B. Ivorra · A. M. Ramos
Instituto de Matemática Interdisciplinar (IMI) & Dept. de Matemática Aplicada, Universidad Complutense de Madrid, Plaza de las Ciencias, 3, 28040 Madrid, Spain
E-mail: ivorra@mat.ucm.es, angel@mat.ucm.es

1 Introduction

In the last decades, consumers are more aware about their diet and they tend to choose healthier food that does not contain additives and with preserved organoleptic (e.g., taste, texture) and nutritional properties. In this context, High-Pressure (HP) processing is one of the most popular food treatments devoted to increasing the shelf-life of this kind of products without damaging their good properties and avoiding the use of additives [1,2].

In HP technology, the packed food is introduced into a vessel that is surrounded by a pressurizing fluid. The pressure of this fluid is incremented and, hence, the pressure of the food also grows, producing some changes in its inner temperature. As a result, the activities of some micro-organisms present in the food are affected by these pressure and temperature variations. Therefore, the design of an HP-Temperature (HPT) treatment, meaning the determination of suitable fluid pressure and control temperatures applied to the food by the equipment, is a non-trivial task. In fact, it can lead to many unsuccessful tests and, consequently, a waste of time and money, each time the food quality requirements change. To assist the food engineers in the configuration of the HP equipment, we propose a decision tool based on solving multi-objective optimization problems. The considered objective functions are the maximum temperature reached in the food during the whole treatment and the final enzymatic activity for their minimization and the final vitamin activity for its maximization. Notice that they are confronted objectives: the larger the temperatures are, the more the enzymatic activity is reduced, but also the more the vitamin activity is decremented. Optimizing all of those objective functions simultaneously consists in obtaining a set of good trade-off points in the search space. This set of compromise individuals is known as Pareto optimal set and its corresponding set of objective vectors is called the Pareto optimal front.

For computing the Pareto optimal set, two types of algorithms can be distinguished. On the one hand, the exact general algorithms (such as the interval branch and bound methods [3]) which can be applied only to solve small problems due to their high cost in time and memory. On the other hand, the meta-heuristics methods provide an approximation of the set of solutions and they are more suitable for complex problems involving many variables and objectives. Among the meta-heuristic methods available in the literature, there are some classical evolutionary multi–objective (EMO) algorithms: SPEA2 [4], NSGA-II [5], MOEA/D [6, 7] and SMS-EMOA [8]. They start from an initial population randomly generated which evolves toward the final set of solutions. In the last decades, methods introducing some preferences of the decision maker (DM) into EMO algorithms have been proposed: the so-called preference-based multi–objective evolutionary algorithms (PMOEAs). Some examples are: WASF-GA [9], and variations of NSGA-II algorithm like R-NSGA-II [10] and its modified versions [11,12].

In this work, we have chosen to use WASF-GA among other evolutionary algorithms because it provides well-distributed results for optimizing problems with more than two objectives. Additionally, it allows focusing the search into a region of interest (ROI) near a reference point (RP) fixed by the DM, avoiding the consideration of unbalanced points that are not interesting for our industrial problem.

As shown in a previous work, the approximated Pareto optimal set obtained with WASF-GA provides to the food engineers a set of points which are adequate for solving different quality requirement situations. Thus, considering this set, they only have to select the point that better satisfies the industry demands at each moment, avoiding the necessity of executing a new optimization procedure each time the quality thresholds change. However, to support this decision tool, the approximated Pareto optimal set must have enough individuals in order to evenly cover the entire ROI. Thus, WASF-GA should handle large population sizes carrying out large computational costs. In particular, for real-world problems, as the one of interest here, it tackles an important challenge due to the fact that the evaluation of the objective function is extremely time-consuming. For instance, in our industrial problem, evaluating the objective function requires to solve numerically a partial differential equations system (that models the heat transfer) coupled with an ordinary differential equation for each enzyme or vitamin involved (describing the evolution of their activities). The number of function evaluations in WASF-GA depends on two input parameters: (i) the number of individuals in the population, and (ii) the number of generations of the optimization procedure. Decreasing their values, trying to reduce the computational cost, can have negative consequences on the performance of the optimization algorithm. In this work, we aim to parallelize the studied optimizer WASF-GA to accelerate its execution and, hence, to take benefit from modern high-performance computers. Broadly speaking, the parallelization is based on the distribution of the function evaluations during the optimization procedure. To analyze the parallel method, a computational study has been carried out, mainly by varying those two parameters which influence the number of function evaluations as well as the number of available processing elements.

This work is organized as follows. In Sect. 2, the industrial problem is formulated in terms of a multi-objective optimization problem in which the numerical simulation of an HPT treatment is involved. Next, in Sect. 3, a parallel methodology for solving it based on the evolutionary algorithm WASF-GA is described. Then, in Sect. 4, the considered numerical experiments are detailed and their main results are analyzed. Finally, in Sect. 5, the conclusions are highlighted.

2 Industrial problem

When a food treatment is applied in the food industry, the main objective is that the final product satisfies the high-quality requirements established by the country legislation. Thus, the processing device, in our case the HPT equipment, must be correctly configured such that the treatment effects on the food properties would be the suitable ones.

In this industrial context, our goal is to provide a decision tool that allows to determine the optimal configuration of an HPT equipment in terms of pressure and temperature according to the final level of enzymatic and vitamin activities and maximum temperature during the process.

The first idea was to formulate a mono-objective optimization problem managing the quality levels as different constraints. However, as studied in a previous work, better results are obtained by considering a multi-objective formulation. Additionally, the multi-objective solution provides a set of points which have been

observed to be suitable for solving different quality requirements situations. Thus, among all these points, food engineers can find configurations that satisfy the demands at different scenarios avoiding the execution of a new optimization procedure each time the quality thresholds change.

Therefore, the problem concerning us is expressed as follows:

$$\begin{cases} \min\ f_{\text{bsaa}}(T_0, T_r, \Delta P_1, \ldots, \Delta P_n), \\ \max\ f_{\text{vit}}(T_0, T_r, \Delta P_1, \ldots, \Delta P_n), \\ \min\ f_{\text{Tmax}}(T_0, T_r, \Delta P_1, \ldots, \Delta P_n), \\ \text{s.t.}\ \ T_0, T_r \in [10, 50](^\circ\text{C}), \\ \qquad \Delta P_1, \ldots, \Delta P_n \in [-250, 250](\text{MPa}), \end{cases} \tag{1}$$

where the objective functions f_{bsaa} and f_{vit} are the enzymatic and the vitamin activities at the final time t_f of the treatment process, respectively, and f_{Tmax} is the maximum temperature reached in the food sample during the HPT process.

Regarding the decision variables of the problem, we have described the configuration of the HPT equipment by means of its initial and refrigeration temperatures, denoted by T_0 and T_r, respectively, and the pressure applied at each instant of the processing time, $P(t)$. This pressure profile is discretized considering a finite number of time steps. The pressure at the initial time is set to the atmospheric pressure, $P_0 = 0.1$ MPa, and the pressure at the next times $P_i, i = 1, \ldots, n$, is the result of adding the value of ΔP_i to the pressure P_{i-1}.

Given a vector formed by these decision variables, the objective functions for the enzyme and the vitamin are computed as the average of the final activity in the food sample domain Ω_{F} at time t_f:

$$f_j(T_0, T_r, P) = \frac{1}{|\Omega_{\text{F}}|} \iint_{\Omega_{\text{F}}} A_j(r, z, t_f) \mathrm{d}r\mathrm{d}z,$$

where $A_j(r, z, t)$ is the activity of a particle located at a point $(r, z) \in \Omega_{\text{F}}$ (see Figure 1) at time t. The index j is used to distinguish between the enzyme and the vitamin activities.

As explained in [13], the activity $A_j(r, z, t)$ is calculated as:

$$A_j(r, z, t) = A_j(r, z, 0) \exp\left(- \int_0^t \kappa_j(P(\sigma), T(r, z, \sigma)) \mathrm{d}\sigma \right), \tag{2}$$

where $\kappa_j(P, T)$ is the inactivation rate (s^{-1}) modelled by a combination of Arrhenius and Eyring equations (see, e.g., [14]) in order to take into account their pressure P and temperature T dependencies:

$$\kappa_j(P, T) = \kappa_{\text{ref}} \exp\left(-B \left(\frac{1}{T} - \frac{1}{T_{\text{ref}}} \right) \right) \exp\left(-C(P - P_{\text{ref}}) \right).$$

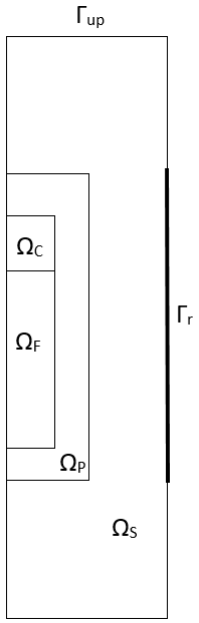

Fig. 1 Computational domain $\Omega = \Omega_{\mathrm{F}} \cup \Omega_{\mathrm{C}} \cup \Omega_{\mathrm{P}} \cup \Omega_{\mathrm{S}}$.

Therefore, we need to simulate the whole HPT process coupling the activity equations given in (2) with the following heat transfer system:

$$\begin{cases} \rho C_p \dfrac{\partial T}{\partial t} - \dfrac{1}{r}\dfrac{\partial}{\partial r}\left(rk\dfrac{\partial T}{\partial r}\right) - \dfrac{\partial}{\partial z}\left(k\dfrac{\partial T}{\partial z}\right) = \alpha \dfrac{\mathrm{d}P}{\mathrm{d}t} T \text{ in } \Omega \times (0, t_{\mathrm{f}}), \\ k\dfrac{\partial T}{\partial \mathbf{n}} = 0 \qquad \text{on } \Gamma \backslash (\Gamma_{\mathrm{r}} \cup \Gamma_{\mathrm{up}}) \times (0, t_{\mathrm{f}}), \\ k\dfrac{\partial T}{\partial \mathbf{n}} = h(T_{\mathrm{amb}} - T) \qquad \text{on } \Gamma_{\mathrm{up}} \times (0, t_{\mathrm{f}}), \\ T = T_{\mathrm{r}} \qquad \text{on } \Gamma_{\mathrm{r}} \times (0, t_{\mathrm{f}}), \\ T(0) = T_0 \qquad \text{in } \Omega. \end{cases} \tag{3}$$

This heat transfer system receives the temperatures T_0 and T_r and the pressure P applied to the equipment at certain time steps defined by the parameters $\Delta P_1, \ldots, \Delta P_n$ described previously. Then, it allows to compute the evolution of the temperature T in the whole domain Ω (see Figure 1). A detailed description of the mathematical model, the computational domain and the boundary conditions considered can be found in [13].

The temperature evolution is also needed by the third objective function, which corresponds to the maximum temperature reached in the food sample domain

during the whole HPT process:

$$f_{\mathrm{Tmax}}(T_0, T_r, P) = \max_{(r,z)\in\Omega_{\mathrm{F}},\ t\in\left[t_0,t_f\right]} T(r,z,t).$$

3 The parallelization of the optimization algorithm: ParWASF-GA

The sequential optimization algorithm initially used to solve Problem 1 is the Weighting Achievement Scalarizing Function Genetic Algorithm (WASF-GA). Broadly speaking, WASF-GA is based on a Genetic algorithm (GA) that applies procedures inspired from the Darwinian natural evolution on a pool of candidate solutions, called population. It works by iteratively modifying the population through the application of genetic operators such as crossover and mutation. Its distinctive feature comes from the selection procedure. Basically, it consists in a mechanism to promote the survival of individuals which are located in the region of interest.

For selecting the individuals that will be maintained in the population at next iteration, WASF-GA considers an achievement scalarizing function (ASF), known as Wierzbicki's ASF [9]. For each individual in the population the ASF is calculated for different weights vectors $\mu^j = (\mu_1^j, \ldots, \mu_m^j)$, where $\mu_i^j > 0$ and $i = 1, \ldots, m$, with $j = 1, \ldots, N_\mu$. Attending to those values, the individuals are classified into different fronts as follows: For each weight vector, the individual with the lowest ASF value is selected, copied to the first front and deleted from the list. It means that such individuals cannot be considered anymore. This procedure is repeated until the first front includes N_μ individuals, one minimizing each ASF. For filling up the following fronts, the same procedure is carried out. The population at the next iteration is composed by the N first selected individuals belonging to the first fronts, where N is an input parameter provided by the DM representing the number of solutions in the ROI. For the problem at hand, $N = N_\mu$. Hence, the next population coincides with the first front. More details about this method and other procedures of WASF-GA can be found in [9].

Algorithm 1: Pseudo-code of the ParWASF-GA optimizer

Input: **q**, **f**, $\{\mu^1, \ldots, \mu^{N_\mu}\}$, N, $h_{\max}$
Output: IndividualSet *population*

```
1  IndividualSet population[2 * N];                    /* Shared among threads */
2  ThreadPool threads = createThreads();               /* Create a set of threads */
3  threads.runInParallel();                            /* Thread-local below: */
4  Load range = getPieceWork();          /* Obtain the work zone for each thread */
5  population = CreateInitialPopulation(range.pop)
6  < barrier > ;   /* To prevent accesses to the population before it is ready */
7  for h = 1 to h_max do
8  |   IndividualSet parents = SelectParents(population, range.pairs);
9  |   IndividualSet offspring = Crossover(parents);
10 |   IndividualSet offspringMut = Mutation(offspring);
11 |   population = Include(population, offspringMut, range.write);
12 |   population < single − barrier >= Selection(population);
13 end
14 return population[1 − N]
```

The parallel version of WASF-GA, called ParWASF-GA throughout this paper, has been designed to be executed in a shared-memory environment. A brief scheme of ParWASF-GA is shown in Algorithm 1. It is based on the fact that the evaluation of each individual in the population is independent from the remaining ones and is computationally intensive. Therefore, an intrinsic parallelism can be accomplished by distributing the evaluation of individuals among threads when the procedures as initialization (line 5), crossover (line 9) and mutation (line 10) are executed. Concerning the selection procedure (line 12), notice that it has not been parallelized to be consistent with the sequential version, i.e. it is necessary to know the whole set of individuals to be able to classify them into fronts on similar way as done by the sequential version. Additionally, its computational cost is negligible, and then its parallelization in terms of efficiency may be counter-productive (it is more than 1200 times faster than a single function evaluation). Notice that the parallel work-flow includes two barriers. The first one (lines 6) guarantees that the initial population is fully completed before any thread can access to it. The second one (line 12) ensures that the whole offspring is included in the population before to proceed with the execution of the selection mechanism, which is carried out by a single thread. Finally, the procedures $createThreads$, $runInParallel$ and $getPieceWork$ (lines 2-4) create a set of threads to work on different ranges of the population list.

As its sequential counterpart, ParWASF-GA requires five input parameters (see Algorithm 1). The region of interest (ROI) should be indicated by the decision maker through a reference point $\mathbf{q} = (q_1, \ldots, q_m)$ formed by a preference value q_i for each one of the objective functions f_i, with $i = 1, \ldots, m$. A sample of weight vectors $\{\mu^1, \ldots, \mu^{N_\mu}\}$ should be provided. Each weight vector has a number of components equal to the number m of objective functions, i.e., $\mu^j = (\mu_1^j, \ldots, \mu_m^j)$ and $\mu_i^j > 0$ for $j = 1, \ldots, N_\mu$ and $i = 1, \ldots, m$. At each generation, $\mathbf{q}$ is projected onto the optimal front in the direction given by the inverses of the weights. Finally, the number $h_{\max}$ of generations and the number N of individuals in the population, at each generation, must also be provided by the DM. The total number of function evaluations consumed is equal to $N \cdot h_{\max}$.

4 Computational experiments and results

The numerical experiments were performed on one single cluster node with shared-memory called Bullion belonging to the infrastructure of the High Performance Computing and Algorithms research group (`http://www.hpca.ual.es/en/`). This node has 8 Intel Xeon E7 8860v3 3.2 GHz processors with 16 cores each one and 2.3 TB of RAM. Despite the available computational resources, the computational analysis has been carried out with up to 32 cores, mainly due the high computational effort associated to the problem at hand. Notice that the optimization algorithms are heuristics, and hence, each particular instance has been solved 5 times to compute average values.

4.1 Considered experiments

Now, we describe the industrial problem considered during this work. The HPT equipment consists of a cylindrical vessel similar to the pilot unit (ACB GEC Alsthom, Nantes, France), whose detailed description, dimensions and thermophysical properties can be found at [13, 15]. Moreover, the pressurizing fluid used during the simulated HPT treatment was liquid water and the considered solid food sample was tylose [16], which is a methyl cellulose gel similar to meat. Their thermophysical properties were described depending on pressure and temperature by means of the shifting approach explained at [17].

Focusing on the micro-organisms whose activity is studied, we consider the enzyme called *Bacillus Subtilis* α*–Amylase* (BSAA) and the vitamin C. The former, BSAA, is produced by a bacteria that is present in the ground. We are interested in decreasing its activity because it contaminates food and it can even modify some organoleptic properties. On the other hand, the vitamin C, which is contained in some fruits, is well-known for its antioxidant properties, so we want to preserve it. The parameters needed for modelling their inactivation rates can be consulted in [14] and [18], respectively.

Finally, a processing time of $t_f = 900$ seconds was set and eight parameters $\Delta P_1, \cdots, \Delta P_8$, each one between -250 and 250 MPa, are considered in order to determine the pressure variations applied to the equipment during this time.

The numerical simulations of the HPT processes are computed using the commercial software COMSOL Multiphysics 5.3. COMSOL solves numerically that kind of partial differential equation systems employing the Finite Element Method (FEM), which, broadly speaking, is based on the approximation of the unknown fields at each point of the spatial mesh covering the whole computational domain by polynomials of a determined degree [19]. More precisely, in this work, P2 Lagrange Finite Elements and quadratic discontinuous Lagrange shape functions have been considered for the temperature in System (3) and for the enzyme and vitamin activities in Equations (2), respectively. On the other hand, the time integration is performed using the BDF method available in this software. For solving the nonlinear systems, a damped Newton method is employed and the algebraic linear systems are solved using MUMPS, which is the acronym for MUltifrontal Massively Parallel sparse direct Solver.

The reference point used here is $(0.0, 0.0, 30.0)$, which corresponds to the total elimination of the enzymatic activity, the complete conservation of the vitamin activity and a maximum temperature reached during the whole process of 30°C.

The performance of the heuristic optimization methods, as the ones considered in this work, highly depends on the selected input parameters (see Sect. 3). In the current scenario, where the evaluation of the objective function is very time demanding (around 47 seconds), adjustment of the input parameters might be oriented to achieve a trade-off between runtime and quality instead of granting optimality. The computational effort of WASF-GA mainly depends on the number of generations ($h_{\max}$) and the number of individuals in the population (N). Notice that the total number of function evaluations is equal to $h_{\max} \cdot N$. The remaining input parameters have influence in the effectiveness of the optimization method, but they do not affect both the effectiveness and efficiency simultaneously. Then, they are out of the scope of this study.

Table 1 Performance of both WASF-GA and ParWASF-GA for $N = 200$ and three different values of $h_{\max}$.

$h_{\max}$	WASF-GA		ParWASF-GA				
	$Av(H)$	$Av(Time)$ (hours)	$Av(Sp)$				
			2	4	8	16	32
10	0.52	25.80	2.00	3.82	7.29	14.14	22.68
20	0.55	54.19	2.00	3.87	7.35	14.16	22.68
30	0.56	82.87	2.00	3.90	7.37	14.17	22.69

In the next subsection, we will focus on how the optimization methods behave for different values of $h_{\max}$ and N.

4.2 Obtained results

Table 1 shows for a fixed value of $N = 200$, the mean hypervolume $Av(H)$ and the average execution time $Av(Time)$, in hours, obtained by WASF-GA for three different values of $h_{\max}$. The hypervolume is a metric frequently used to measure the effectiveness of a multi-objective optimization method [20–23]. Broadly speaking, the larger its value is, the better the results are. Additionally, the speedup $Av(Sp)$ obtained by ParWASF-GA with up to 32 threads is shown. As it can be observed in this table, we are able to significantly increase the quality of the results by increasing $h_{\max}$ (from 0.52 when $h_{\max} = 10$ to 0.56 for $h_{\max} = 30$). The overall runtime is also higher, though. Hence, parallelization is of major interest. As expected, the efficiency of ParWASF-GA is not affected by $h_{\max}$, since this parameter just scales the overall computing time. Furthermore, the speedup obtained by ParWASF-GA is almost linear for up to 16 threads, achieving a reasonable mean value of 14.17. We note that the hypervolume obtained by ParWASF-GA is not reported, but it is similar to the one obtained by the sequential version.

Table 2 shows the behaviour of the parallel version for up to 32 threads, with three different population sizes, $N = 50, 100, 200$, and a fixed number of generations, $h_{\max} = 30$. For completeness, the average computing time of the sequential version ($Av(Time)$) has been also included. We can observe in this table that the speedup improves with the problem size. For example, with 32 threads the speedup increases from 14.44 with N=50 up to 22.69 with N=200. Regarding the effectiveness, the hypervolume obtained by ParWASF-GA remains similar to the one achieved by the sequential version, independently of the number of considered threads. Thus, these values have been omitted in this table.

Considering these results, the algorithm proposed, ParWASF-GA, obtains reasonable speedups and shows scalability. Therefore, the use of high-performance computing for optimizing the HPT treatment is recommendable.

Table 2 Performance of both WASF-GA and ParWASF-GA for $h_{\max} = 30$ and three different values of N.

N	WASF-GA	ParWASF-GA				
	$Av(Time)$ (hours)	$Av(Sp)$				
		2	4	8	16	32
50	18.96	2.00	3.83	7.02	11.76	14.44
100	37.32	2.00	3.85	7.14	13.41	20.08
200	82.87	2.00	3.90	7.37	14.17	22.69

4.2.1 On the parallelization of the objective function evaluation

The parallel programming paradigm used to design ParWASF-GA has been mainly motivated by the cost associated to the objective function (around 47 seconds). As previously mentioned, the evaluation of the objective function at a particular decision vector consists in solving numerically the model composed by Equations (2) and System (3). For this task, as said previously, we have used the commercial software COMSOL Multiphysics because it is an exhaustive FEM tool that allows to obtain quickly models based on PDEs.

The considered numerical model corresponding to System (3) is computationally intensive. During its resolution, several linear systems have to be solved. However, solving a linear system is a parallelizable task. Indeed, COMSOL implements a shared-memory parallel strategy such that the entire matrices and vectors are available for all the threads in the common memory and the operations over these matrices and vectors are distributed among the threads.

This subsection is devoted to analyze the performance of ParWASF-GA when we exploit the parallel capabilities of COMSOL at the evaluation of the objective function.

Table 3 shows the behaviour of ParWASF-GA for the case with the population size $N = 200$ and the number of generations $h_{\max} = 30$, and where the objective function is also parallelized. More precisely, column $Av(Time\_p)$ refers to the computing time consumed by the sequential algorithm WASF-GA, when the objective function, which has been implemented in COMSOL, is executed by using p threads. Columns 3 to 6 indicates the speedup obtained by ParWASF-GA with respect to $Av(Time\_p)$. Then, row 1 refers to the speedup of ParWASF-GA when the parallelization of the objective function is not carried out; while row i, with $i > 1$, indicates the corresponding speedup when the objective function has been executed by using i threads. Having a look to the results, one can infer that is not recommendable to parallelize the objective function by using more than 2 threads. In fact, considering more than 4 threads is completely inadvisable; i.e. the computing time with $p = 4$ is equal to 65.08 hours, while it increases to 70.56 hours with $p = 8$ (see column $Av(Time\_p)$). This is an interesting result for the COMSOL users, which may intuitively think that the parallelization of the objective function may be enough to accelerate an optimization method, as the one considered in this work. This phenomenon has been yet observed in other works proposed in the literature, for instance when optimizing the trajectory of oil spill cleaning ships paralellizing the considered GA was much more efficient than parallelizing the cost function, see [24,25].

Table 3 Performance of WASF-GA and ParWASF-GA when the number of processors devoted to COMSOL varies.

	$Av(Time\_p)$ (hours)	Speedup			
COMSOL \ WASF-GA	1	2	4	8	16
1	82.87	2.00	3.90	7.37	14.17
2	67.50	2.00	3.57	7.00	12.45
4	65.08	1.89	3.60	7.20	11.91
8	70.56	1.91	3.95	6.95	12.88

To sum up, in view of the results, it is much more advisable to perform the parallelization of the optimization method. In fact, our proposed ParWASF-GA seems to behave good independently of the number of threads used to parallelize the objective function, i.e. the obtained speedup is close or equal to the ideal one.

5 Conclusions

In this work, we consider a multi-objective problem to improve the efficiency of a HPT food treatment. To solve this problem, a parallelized version of the optimization algorithm WASF-GA is proposed. Such a method, called ParWASF-GA allows reducing significantly the computational time. Thanks to this, we can solve our industrial problem considering a larger number of evaluations, which translates on (i) an increase in the number of optimal points provided to food engineers, since larger populations can be considered, and (ii) an increment of the method accuracy, since the optimization algorithm can evolve during a larger number of iterations. A computational study shows that the performances of ParWASF-GA are reasonable, i.e. the speedup is almost linear when there is enough computational load. Additionally, it is scalable: the larger the size of the problem to solve, the better the performance of the parallel version. Finally, a numerical study exhibits that the performance of ParWASF-GA is still quite good when the numerical system involved at the evaluation of the objective function is solved using also several processors. In the future, other parallel programming strategies will be implemented and analyzed.

Acknowledgements This research has been funded by grants from the Spanish Ministry of Economy and Competitiveness (TIN2015-66680-C2-1-R and MTM2015-64865P); Junta de Andalucía (P11-TIC7176 and P12-TIC301), in part financed by the European Regional Development Fund (ERDF). Juana López Redondo is a fellow of the Spanish "Ramón y Cajal" contract program, co-financed by the European Social Fund. Savíns Puertas-Martín is a fellow of the Spanish "Formación del Profesorado Universitario" program, financed by the Ministry of Education, Culture and Sport. The authors gratefully acknowledge Nicolás Calvo Cruz for its suggestions and kind assistance in the redaction of this paper.

References

1. R. Hayashi, Y. Kawamura, T. Nakasa, and O. Okinaka. Application of high pressure to food processing: pressurization of egg white and yolk, and properties of gels formed.

Agricultural and Biological Chemistry, 53(11):2935–2939, 1989.
2. N. K. Rastogi, K. S. M. S. Raghavarao, V. M. Balasubramaniam, K. Niranjan, and D. Knorr. Opportunities and challenges in high pressure processing of foods. *Critical reviews in food science and nutrition*, 47(1):69–112, 2007.
3. J. Fernández and B. Tóth. Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods. *Computational Optimization and Applications*, 42(3):393–419, 2009.
4. E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, 2001.
5. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
6. Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.
7. H. Li and Q. Zhang. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):229–242, April 2009.
8. N. Beume, B. Naujoks, and M. Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.
9. A. B. Ruiz, R. Saborido, and M. Luque. A preference-based evolutionary algorithm for multiobjective optimization: the weighting achievement scalarizing function genetic algorithm. *Journal of Global Optimization*, 62(1):101–129, 2015.
10. K. Deb and J. Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 635–642. ACM, 2006.
11. E. Filatovas, O. Kurasova, and K. Sindhya. Synchronous R-NSGA-II: an extended preference-based evolutionary algorithm for multi-objective optimization. *Informatica*, 26(1):33–50, 2015.
12. E. Filatovas, A. Lančinskas, O. Kurasova, and J. Žilinskas. A preference-based multi-objective evolutionary algorithm R-NSGA-II with stochastic local search. *Central European Journal of Operations Research*, pages 1–20, 2016.
13. J. A. Infante, B. Ivorra, A. M. Ramos, and J. M. Rey. On the Modelling and Simulation of High Pressure Processes and Inactivation of Enzymes in Food Engineering. *Mathematical Models and Methods in Applied Sciences (M3AS)*, 19(12):2203–2229, 2009.
14. S. Denys, L. R. Ludikhuyze, A. M. Van Loey, and M. E. Hendrickx. Modeling Conductive Heat Transfer and Process Uniformity during Batch High-Pressure Processing of Foods. *Biotechnology Progress*, 16(1):92–101, 2000.
15. L. Otero, A. M. Ramos, C. De Elvira, and P. D. Sanz. A model to design high-pressure processes towards an uniform temperature distribution. *Journal of Food Engineering*, 78(4):1463–1470, 2007.
16. T. Norton and D. Sun. Recent Advances in the Use of High Pressure as an Effective Processing Technique in the Food Industry. *Food and Bioprocess Technology*, 1(1):2–34, 2008.
17. L. Otero, A. Ousegui, B. Guignon, A. Le Bail, and P. D. Sanz. Evaluation of the thermophysical properties of tylose gel under pressure in the phase change domain. *Food hydrocolloids*, 20(4):449–460, 2006.
18. L. Verbeyst, R. Bogaerts, I. Van der Plancken, M. Hendrickx, and A. M. Van Loey. Modelling of vitamin C degradation during thermal and high-pressure treatments of red fruit. *Food and Bioprocess Technology*, 6(4):1015–1023, 2013.
19. A. M. Ramos. *Introducción al análisis matemático del método de elementos finitos*. Editorial Complutense, 2012.
20. J. L. Redondo, J. Fernández, and P. M. Ortigosa. Femoea: a fast and efficient multi-objective evolutionary algorithm. *Mathematical Methods of Operations Research*, 85(1):113–135, 2017.
21. J. L. Redondo, J. Fernández, J. D. Álvarez, A. G. Arrondo, and P. M. Ortigosa. Approximating the pareto-front of a planar bi-objective competitive facility location and design problem. *Computers & Operations Research*, 62(0):337 – 349, 2015.
22. L. While, L. Bradstreet, and L. Barone. A fast way of calculating exact hypervolumes. *IEEE Transactions on Evolutionary Computation*, 16(1):86–95, 2012.

23. E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms - a comparative case study. In A. E. Eiben, editor, *Parallel problem solving from nature V*, pages 292–301, Amsterdam, September 1998. Springer-Verlag.
24. C. Alavani, R. Glowinski, S. Gomez, B. Ivorra, P. Joshi, and A. M. Ramos. Modelling and simulation of a polluted water pumping process. *Mathematical and Computer Modelling*, 51(5):461–472, 2010.
25. S. Gomez, B. Ivorra, and A. M. Ramos. Optimization of a pumping ship trajectory to clean oil contamination in the open sea. *Mathematical and Computer Modelling*, 54(1):477–489, 2011.