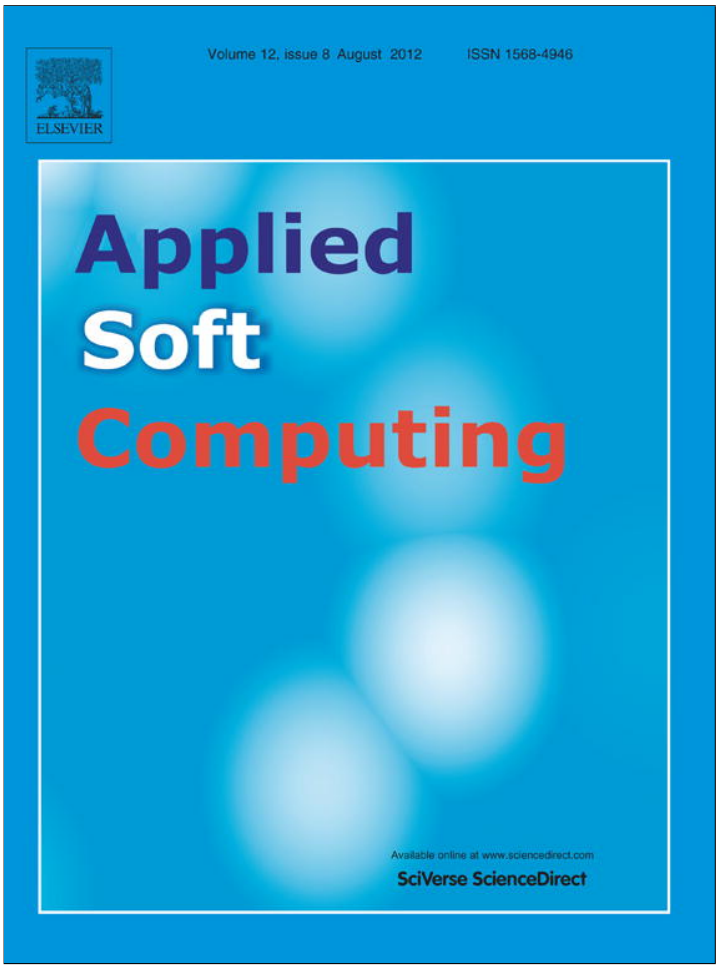

Applied Soft Computing 12 (2012) 2040–2052

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Metaheuristic approaches to grouping problems in high-throughput cryopreservation operations for fish sperm

T.W. Liao [a,*], E Hu [b], T.R. Tiersch [b]

[a] Department of Mechanical and Industrial Engineering, Louisiana State University, Baton Rouge, LA 70803, United States
[b] Aquaculture Research Station, Louisiana Agricultural Experiment Station, Louisiana State University Agricultural Center, Baton Rouge 70803, United States

ARTICLE INFO

Article history:
Received 1 September 2010
Received in revised form 1 March 2012
Accepted 4 March 2012
Available online 17 March 2012

Keywords:
Heuristic
Metaheuristic
Simulated annealing
Tabu search
Ant colony optimization
Grouping problem
Hybrid metaheuristic
Differential evolution
Threshold accepting
High-throughput cryopreservation
Blue catfish

ABSTRACT

High-throughput cryopreservation operations of fish sperm is a technology being developed by researchers today. This paper first formulates a grouping problem in high-throughput cryopreservation operations of fish sperm and then develops a heuristic and four metaheuristic algorithms for its solution. The heuristic is modified from one originally proposed for the assembly line balancing problem. The four metaheuristic algorithms include simulated annealing (SA), tabu search (TS), ant colony optimization (ACO), and a hybrid differential evolution (hDE). For each metaheuristic algorithm, four different initialization methods were used. For both SA and TS, five different neighborhood solution generation methods were also studied. Real world data collected from a high-throughput cryopreservation operation was used to test the effectiveness of algorithms with different initialization and neighborhood solution generation methods. For comparison, a base line of grouping by processing order was also established. The results indicate that: (i) all algorithms performed better than the base line; (ii) using the result of the modified heuristic as the initial solution of metaheuristic algorithms lead to a better solution; the amount of improvement varied from algorithm to algorithm; (iii) among the five neighborhood solution generation operators, insertion operator was the best; (iv) among all algorithms tested, the hybrid differential evolution is the best, followed by tabu search in terms of average objective value.

1. Introduction

Spermatozoa cryopreservation is a practical field in cryobiology research. Its goal is to maintain the biological functions of sperm cells for long periods (e.g., years or decades) by storage at cryogenic temperatures (<−190 °C). This technology has been applied in mammals, birds, amphibians, fishes, and invertebrate species and includes medical application, livestock breeding, endangered species preservation, and genetic research [1]. Starting in the late 1940s and early 1950s, a series of discoveries were published for mammalian sperm cryopreservation [2,3]. Since then, the application of cryopreserved sperm in livestock has expanded into a multibillion dollar global industry [4], and has changed breeding methods worldwide.

Fish sperm cryopreservation was demonstrated at the same time as for humans and livestock species [5], however, despite the large-scale application in mammalian species, cryopreservation remains at a research scale for aquatic species. During the past decades, there have been >200 aquatic species reported for sperm cryopreservation [6] and this number is increasing. However, high-throughput processing has not yet become available for commercial-scale application.

The channel catfish (*Ictalurus punctatus*) is the basis of the largest foodfish aquaculture industry in the United States [7]. Hybrid catfish, created by crossing of female channel catfish and male blue catfish (*I. furcatus*) are in high demand by the industry because of their fast growth rate and efficient food conversion [8]. Recently, a survey of fish farmers revealed a strong demand for the kinds of genetic improvement that can be provided for commercial use by cryopreserved sperm [9], especially among hybrid catfish farmers. However, blue catfish do not readily hybridize with channel catfish naturally [8], and the availability of blue catfish is often reliant on wild caught (rather than farm-raised) fish. Sperm cryopreservation is required for preserving sperm samples from valuable male blue catfish, and to provide sperm for hybrid breeding at times when female channel catfish are in peak reproductive condition. To date, sperm cryopreservation of blue catfish has been studied in the laboratory [10] and for preliminary adoption of the commercial approaches used for dairy bulls [11], and a basic protocol based on mammalian methods has been generated. However, to meet the

* Corresponding author at: Department of Mechanical and Industrial Engineering, Louisiana State University, 2508 P.F. Taylor Hall, Baton Rouge, LA 70803, United States. Tel.: +1 225 578 5365.
E-mail address: ieliao@lsu.edu (T.W. Liao).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2012.03.007

large product demand necessary for commercial hybrid catfish production, an approach for large-scale production of cryopreserved sperm is needed.

High-throughput cryopreservation has been widely applied in the dairy industry for decades. Based on the procedures developed in the laboratory, high-throughput cryopreservation processes were developed for increased processing speed, capability for mass production, and quality assurance. In a previous study that used dairy sperm protocols for catfish [11], technical feasibility of the approach was demonstrated, but the need for facilities dedicated to aquatic species cryopreservation became apparent. Therefore current research is focused on the development and improvement of a high-throughput sperm cryopreservation approach for aquatic species, in this case for blue catfish, in coordination with development of a production process designed specifically for use with fish and shellfish. An automated loading, sealing, labeling and reading system named MAPI (developed by CryoBioSystem Inc., Paris, France) has been utilized for high-throughput mammalian sperm cryopreservation. Adaptation of this equipment has been evaluated for aquatic species to enable commercial-scale use of cryopreserved sperm [12] and comprises four major steps: fish selection, sperm collection and processing, straw packaging and freezing, and thawing and use (Fig. 1). These steps are briefly described below:

Fish selection - mature blue catfish are collected from holding areas (pond or tanks), and are selected for reproductive traits and transported to the cryopreservation facility.
Sperm collection and processing – the most promising males are killed and the testes are removed by dissection and rinsed with Hanks' balanced salt solution (HBSS). Adherent tissues and blood are cleaned and the testis is rinsed with HBSS. Cleaned testes are placed in Ziploc bags and HBSS is added in a 1:2 ratio (testis wight (g): HBSS volume (ml)). The air is pushed out before sealing the bags. The testes are crushed by application of pressure until all pieces become transparent. The suspension from the Ziploc bags is passed through 200-μm filters and the cell concentration is assessed by spectrophotometer or hemocytometer. The final sperm concentration is adjusted to 1×10^9 cells/ml. Every male is processed individually with biological information recorded. Necessary steps such as labeling are taken to prevent contamination among samples.

The sperm suspensions and corresponding biological information are gathered before proceeding to the MAPI system. The number of straws required is estimated from the volume of the sperm suspensions, referring to Eq. (10) for blue catfish. Based on the estimated numbers of straws, the suspensions must be organized into several groups in consideration of the capacity of automated system, and freezer, post-freeze handling, and storage method, and straw production proceeds as sequential groups.

Straw packaging and freezing (production in groups) – sufficient methanol is added to working solutions to yield a 10% final concentration (v:v) as a cryoprotective agent. The mixtures are loaded into 0.5-ml proprietary plastic (CBS) straws by the MAPI system and each straw is labeled with alphanumeric and bar-coded tracking information. All samples within the group have to be packaged within 30 min [12]. Between samples, there will be a setup time for placing new sample, replacing injection nozzle, and typing label. The straws leaving the MAPI system are transferred directly to a programmable freezer (Micro Digitcool, IMV, France). The freezing process is from 4 °C to −80 °C with cooling rate of 5 °C/min. After freezing, the straws are placed into 12-compartment goblets (plastic containers) and stored under liquid nitrogen at −196 °C. The current freezer used has the capacity of 240 straws.

Sample usage – based on the demand for sperm samples, the specified number of straws are removed from inventory and shipped to the destination in vapor-phase liquid nitrogen shipping dewars. After arrival, the straws are taken out of the dawer and thawed immediately at 40 °C for 20 s. The thawed straws are used for analysis or application in fish production.

The grouping process between Steps 2 and 3 described above is critical to the entire high-throughput process because it directly affects the operation efficiency and thus cost. If the number of groups is too high, there will be too many production cycles; if the groups are not well balanced, the number of straws packaged in each cycle will vary affecting final quality (cell viability) of the samples and overall throughput. In real life application, when there were more than 10 samples ready at the same time, the operator had to stop the process for manual grouping often by processing order, which significantly affected the working efficiency and production.

The goal of this study was two-fold: first to formulate the grouping problem for the above-described high-throughput cryopreservation operations, and to develop one heuristic and four metaheuristic algorithms for its solution. The high-throughput process is new and the grouping problem does not resemble any existing problems addressed in the production and operation management literature. One reviewer pointed out that the subject application could be considered as a one-dimensional bin-packing problem. After further consideration of the process, the grouping problem really does not fit the one-dimensional bin-packing problem or its variants due to its unique features, to be detailed in the following section. Among the four algorithms, the hybrid differential evolution algorithm is newly developed and the other three metaheuristic algorithms themselves are not new; but tailoring them to this new application is.

This paper is organized as follows: Section 2 introduces the mathematical formulation of the problem. Section 3 presents the heuristic algorithm. Section 4 describes the four metaheuristic algorithms developed and tailored to the subject application. Section 5 explains collection of the test data and presents the test results. Discussion follows in Section 6, and Section 7 concludes the paper.

2. Problem formulation

A review of the operational research and production and operation management literature reveals that the grouping problem addressed in this report does not match exactly to any existing problem. One reviewer pointed out that the subject application could be considered as a one-dimensional bin-packing problem. In the followings, some unique features of the grouping problem are described to indicate otherwise. First of all, all straws can be produced from each particular fish must be labeled uniquely. There is thus a setup time associated with each fish on the MAPI system. Secondly, the number of straws can be produced a fish varies greatly from one fish to another and from one season to another. Thirdly, there are two constraints associated with the grouping problem. One is the equilibrium time required to complete one batch (group) of MAPI operation that is 30 min for the blue catfish studied in this paper. Another constraint is the capacity of the freezer that is 240 straws for the one being used. Depending upon the situation, either one of the above two constraints might dominate over the other one.

Let $\mathbf{X} = \{x_1, x_2, \ldots, x_i, \ldots, x_n\}$ be a vector of estimated numbers of straws for n fishes to be processed. The task is to from organized groups subject to the two constraints mentioned above, i.e., the constraint of MAPI equilibrium time, T_{MAPI}, and the freezer capacity

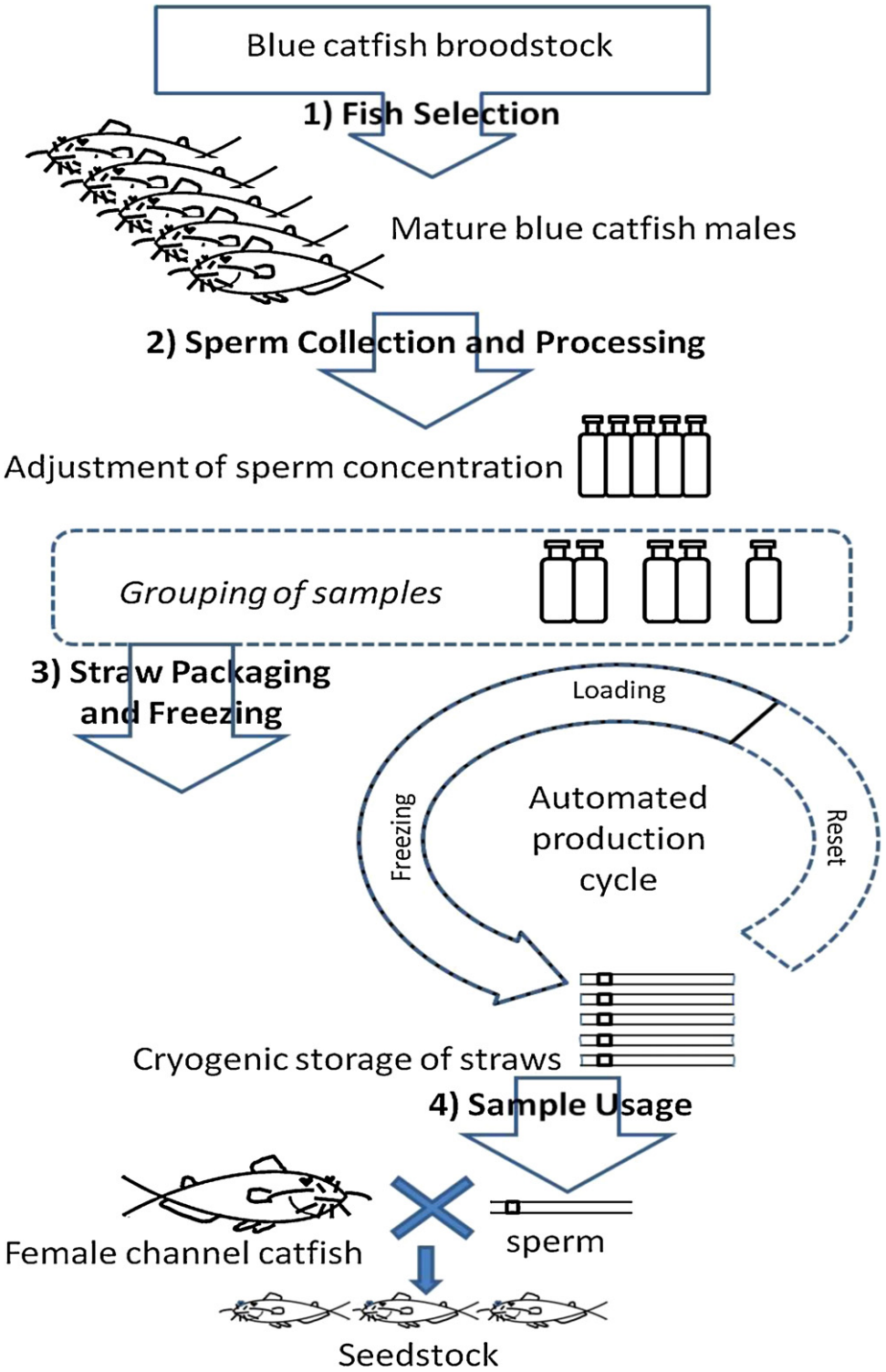

Fig. 1. A sketch of operations involved in high throughput cryopreservation of blue catfish sperm.

constraint, C. In essence, the high-throughput cryopreservation system is constrained mainly by the MAPI system and the freezer. Whichever dominates will be the capacity constraint for the grouping problem. If the number of straws can be produced from a fish is low, then more fishes are needed to form a group to better utilize the full freezer capacity. However, the setup time will also be high because each fish requires the preparation of a unique label. There is no precedence relationship among the fish providing samples. One unique situation is that an individual fish could provide more straws than the capacity constraint. To deal with this situation, each individual x_i is first divided into parts such that each individual part is less than C. This can be done in several ways. For this study, we consider $x_i = k\ C + remainder$. Therefore, the number of parts is $k+1$ with the first k parts equal to C and the last part as the remainder.

To evaluate the goodness of each grouping, some criteria were needed. Two criteria were considered in this study: efficiency, E (the larger the better) and the imbalance between groups, I (the

smaller the better). With m as the number of groups formed, the efficiency, E, was computed as follows:

$$E = 100\frac{\sum_{i=1}^{n} x_i}{m \cdot C} \quad (1)$$

Note that E increased if the number of groups formed, m, was small.

With S_j as the total numbers of straws in a group j, i.e., $S_j = \sum_{i \in group\, j} x_i$. The imbalance between groups, I, was computed as:

$$I = \sqrt{\sum_{j=1}^{m} (S_j - \max(S_j))}. \quad (2)$$

The grouping problem was formulated:

$$\text{Min} f = w(100 - E) + (1 - w) I \quad (3)$$

$$s.t.\ S_j < C,\ \forall j \quad (4)$$

$$t^*_{setup}|j| + t^*_{straw} S_j < T_{MAPI},\ \forall\ j \quad (5)$$

The weight w in Eq. (3) had a value between zero and one, depending upon the relative importance of each criterion. In this study, w was fixed at 0.5, meaning that both criteria were considered to be equally important. In Eq. (5), t_{setup}, t_{straw}, and $|j|$ denote unit setup time for each fish, unit time for producing one straw, and number of fishes in group j. For the current system, t_{setup} and t_{straw} are 63 s and 60/11 s, respectively.

The total number of unique ways for grouping n fishes is $n!$, which is equal to 2.4329×10^{18} when $n = 20$, 2.6525×10^{32} when $n = 30$, 8.1592×10^{47} when $n = 40$, and 3.3142×10^{126} when $n = 84$. Thus, the numbers of unique sequences explode as the number of fish increases. Due to this large search space, solution of the grouping problem had to rely on heuristics. It will be shown later that metaheuristic algorithms were effective solution methods for the subject problem.

3. A heuristic

A heuristic algorithm originally designed for an assembly line balancing problem by [13] was modified for solving the subject grouping problem. Note that this heuristic was found to outperform five other heuristics in a recent comparative study [14].

The modified Hoffmann heuristic (labeled mH in the sequel) is described below.

1. Start with the full vector, $\mathbf{X} = \{x_1, x_2, \ldots, x_i, \ldots, x_n\}$, assuming that each x_i has been divided into parts; in other words, each x_i is less than C. Let g be zero, indicating that the number of groups formed so far is none.
2. Test each subset in the power set of $\mathbf{X}$ that contains at least one x_i and satisfies Eq. (5), i.e., the MAPI equilibrium time, and compute the associated idle time that is defined as the difference between C and the sum of all elements in the subset.
3. Select the subset that has the minimal idle time and all elements in the subset are assigned to the next group, increment g, and remove the elements in the selected subset from the vector $\mathbf{X}$.
4. Repeat 2–3 till that the vector $\mathbf{X}$ is empty.

This heuristic can be used alone or used to generate the initial solution(s) for metaheuristic algorithms to be introduced in the next section.

Simulated annealing for grouping problem in high throughput cryopreservation operations.

```
Read in the data and set the maximum number of evaluations (max_nfe) to
  be used as the stopping criterion.
Set the maximum temperature (t_max), the cooling rate (α), and the
  maximum number of iterations (i_max) for each temperature.
Generate an initial sequence, convert it into groupings and evaluate its
  performance, f, and set current number of evaluations (nfe) to one.
Let overall best and current best equal to f, i.e., f_opt = f_cur = f and current
  temperature be the maximum temperature, i.e., t_cur = t_max
While nfe < max_nfe
  Let the iteration counter be one, i.e., itr = 1
  While itr < i_max
    Apply a neighborhood search operator to generate a new candidate
  sequence, evaluate its performance, f, and increment nfe by one
    If f < f_cur or exp((f_cur − f)/t_cur) > rand (a randomly generated value
  between 0 and 1)
      Accept the candidate solution and make it as current, i.e., f_cur = f.
    End
    Increment itr by one
  End
  If f_cur < f_best
    Update f_best and the associated solution
  End
  Update the current temperature by cooling rate, i.e., t_cur = α t_cur
End
```

4. Four metaheuristic algorithms for solving the grouping problem

Before presenting the four metaheuristic algorithms, how each solution is represented and evaluated is first described below. Each possible solution is represented as a sequence for all four metaheuristic algorithms. A simple procedure is used to convert a sequence into groupings. The procedure involves taking each element in the sequence by the position order, putting them in the same group, and starting a new group if the capacity constraint is exceeded. A sequence with any group that fails to satisfy the MAPI constraint is considered infeasible and discarded (making it non-competitive by assigning a zero efficiency value, i.e., $E = 0$, hence a high objective function value). The performance of each sequence is a weighted average of two criteria: the efficiency and the imbalance between groups, as described in the previous section.

4.1. Simulated annealing and neighborhood search operators

The simulated annealing (SA) algorithm for optimization originated from the work of Kirpatrick et al. [15], in which it was first shown how the Metropolis algorithm for approximate numerical simulation of the behavior of a many-body at a finite temperature provided a natural tool for bringing the techniques of statistical mechanics to bear on optimization. Its uses in combinatorial optimization problems such as partitioning, component placement, wiring of electronic systems, and traveling salesman problems were described. Since then, simulated annealing has been used in other applications.

A search of the Web of Science database with "simulated annealing" as keywords produces 9231 hits at the time of writing this paper on July 29, 2010, and 10,159 hits at the time of rewriting this paper on Feb. 24, 2010. The effectiveness and popularity of simulated annealing as an optimization method is thus well founded. Simulated annealing was chosen for this study because it is one of the oldest metaheuristics and could be as effective as modern metaheuristic algorithms depending upon the application. To investigate its effectiveness, simulated annealing is thus tailored for the subject application. The pseudo-code for the simulated annealing algorithm implemented in this paper is given below.

The three major parameters associated with simulated annealing are the maximum temperature (t_{max}), cooling rate (α), and

maximum number of iterations (i_{max}) for each temperature. The initial solution (sequence) is generated four different ways: by data order (labeled as 1 in the sequel), randomly generated (labeled 2), ranked by ascending values (labeled 3), and using the modified heuristic described in Section 3 (labeled 4). A neighborhood search operator is needed to generate candidate solutions. To this end, five operators were implemented. The first four neighborhood search operators are paired swap, inversion mutation, shift mutation, and insertion. The fifth operator is random selection of any one of the above four operators.

The paired swap operator involves swapping the values of two randomly chosen unique positions in the sequence. Paired swap is also known as exchange mutation in the genetic algorithms literature. The inversion mutation operator involves the inversion of the substring between the two randomly chosen two positions with the two positions included. The inversion mutation operation is identical to the paired swap operation if the two positions are adjacent or there is only one position between the two points. The shift mutation operator involves randomly choosing a position and making it the first position of the new sequence; the substring before the chosen position in the old sequence is shifted to the end of the new sequence. The insertion operator involves moving the value in the second randomly chosen position to a position before the value of the first randomly chosen position. If the two chosen positions are adjacent, the result of insertion is identical to the result of paired swap.

4.2. Tabu search

The most cited paper on tabu search (TS) is the tutorial written by Glover [16]. The basic tabu search is characterized by the use of a tabu list to prevent revisiting an old solution, an aspiration criterion to accept an old solution in the tabu list, and a way of finding the best admissible solution among a set of trial solutions. A search of the Web of Science database with "tabu search" as keyword produce 2979 hits, at the time of writing this paper on July 29, 2010, and 3475 hits at the time of rewriting this paper on Feb. 24, 2010. Tabu search is the second oldest metaheuristic algorithm and has been shown to be effective for several applications. To investigate its effectiveness, tabu search is also tailored for the subject application.

Using the same solution representation scheme, the same procedure to convert a sequence into groupings, and the same performance evaluation criteria as described in Section 4.1, the pseudo-code of the tabu search algorithm implemented in this paper is given below.

The two major parameters associated with tabu search are the size of tabu list (*tls*), and the number of trial solutions (*nts*). The initial solution was generated in four ways, identical to those described above for simulated annealing. The same set of five neighborhood search operators used in simulated annealing was also used with the tabu search algorithm.

4.3. Ant colony optimization

Ant colony optimization (ACO) is a relatively new metaheuristic proposed by Dorigo [17] based on the foraging behavior of ant colonies. In nature, each ant is a simple creature but collectively a colony of ants has an amazing ability to identify the shortest path from their colony to a source of food through stigmergy which is an indirect form of communication used to coordinate activities. Ants exchange information by depositing chemical substances called pheromones that can be sensed by other ants. Initially, each ant wanders randomly. Once an ant finds a food source, it evaluates the quality of the food and carries some to the nest. While returning, the ant deposits a pheromone trail along the path traveled. The deposited pheromone can guide other ants to the food source.

Tabu search for grouping problem in high-throughput cryopreservation operations.

```
Read in the data and set the maximum number of evaluations (max_nfe) to
  be used as the stopping criterion.
Set the size of tabu list (tls), and the number of trial solutions (nts).
Generate an initial sequence, convert it into groupings, evaluate its
  performance, f, and set number of evaluations (nfe) to one
Let f be current best (f_cur) and overall best (f_best) and initialize the tabu
  list be empty
While nfe < max_nfe
  Apply a neighborhood search operator to generate n number of trial
  solutions, evaluate their performances, and increment nfe by n.
  Sort all trial solutions in the ascending order their performance values.
  Determine the best admissible solution by checking each trial solution
  from the best onward.
  If the trial solution is not in the tabu list
    Accept the trial solution and make it the current solution and update
  the tabu list
  Else
    If the performance of the trial solution is better than the current best
      Accept the trial solution and make it the current solution and update
  the tabu list.
    End
  End
End
```

While an isolated ant moves essentially at random, an ant encountering a previously laid trail can detect and follow, with a tendency, the path which has been marked by concentrated pheromone. The level of tendency, equivalent to the quantity of pheromone deposited, is dependent upon the quality of the food. A high level of pheromone attracts ants to follow the trail with a high probability thus reinforcing the trail with additional pheromone. On the other hand, the pheromone concentration also vanishes with time and less-traveled trails will become less attractive. The whole process is therefore characterized by a positive feedback loop where the probability with which each ant chooses a specific path increases with the number of ants having chosen the same path in the preceding steps. As a result, the ant population and path-traversing process converge to the shortest path from the nest to the food source in a relatively short period of time.

A search of the Web of Science database with "ant colony optimization" as keyword produces 906 hits at the time of writing this paper on July 29, 2010, and 1259 hits at the time of rewriting this paper on Feb. 24, 2010. Ant colony optimization has never been tailored for the current subject application before. Our study is the first attempt. The pseudo-code of ACO implemented in this paper is given below:

The four major parameters associated with ant colony optimization were the number of ants (ants), the pheromone evaporation rate (e), and the weights of pheromone (a) and heuristic information (b). The initial populations were also generated in the four ways, identical to those described for the previous two metaheuristic algorithms. However, ACO does not employ the five neighborhood solution generation operators used in SA and TS. Instead, it relies mostly on pheromone and heuristic information.

4.4. Hybrid differential evolution

The hybrid differential evolution (hDE) algorithm enhances the version of differential evolution algorithm used in Ref. [18] with the tabu list concept and the threshold accepting principle. The tabu list concept is originally used in the tabu search algorithm to prevent repeatedly searching the same area of the search space. Instead of picking the generation best solution for intensification, the best solution not in the tabu list is chosen instead. The threshold accepting (TA) algorithm, proposed by Dueck and Scheuer [19], is very similar to simulated annealing and can be considered a modification of it. Both TA and SA have a way to escape from local minima using some acceptance rule. The essential difference

Ant colony optimization for grouping problem in high-throughput cryopreservation operations.

Read in the data and set the maximum number of evaluations (*max_nfe*) to be used as the stopping criterion.
Set the number of ants (*ants*), the pheromone evaporation rate (*e*), and the weight of pheromone (*a*).
Generate the initial population of sequences, convert each of them into groupings, evaluate their performances, and select the best as overall best (*f_best*), and set current number of evaluations (*nfe*) equal to number of ants.
Initialize the matrix of pheromones to one, i.e., $\tau(i,j)=1, \forall i,j=[1,n]$, where n is the number of elements in the sequence. Compute the heuristic information as the absolute difference between positions in the sequence constructed by the modified Hoffmann heuristic, S_H as follows: $h(i,j)=\text{abs}(i-j)$, where i and j are two positions in the S_H.
While *nfe* < *max_nfe*
For each ant, randomly pick the first fish to group.
For each ant k, select the next fish based on probability computed as follows in order to construct the sequence:

$$P_k(i,j)=\begin{cases}\dfrac{\tau(i,j)^a h(i,j)^b}{\sum_{l\in N_i^k}\tau(i,l)^a h(i,l)^b} & \forall j\in N_i^k\\ 0 & else\end{cases} \quad (6)$$

In Eq. (6), N_t^k is the neighborhood of ant k after selecting fish i; the neighborhood of fish i contains all the remaining fishes that have not been grouped.
Evaluate the performance of each ant solution and increment *nfe* by the number of ants.
Select the best ant solution and the current best (*f_cur*).
If *f_cur* < *f_best*, accept the candidate solution and update *f_best*.
Update the pheromone traces on sequences constructed by all ants and the best ant, additionally by using the following equation:

$$\tau(i,j)=(1-e)\cdot\tau(i,j)+e\cdot f \quad (7)$$

In Eq. (7), f is the performance associated with a relevant sequence.
End

between SA and TA lies in the specific acceptance rules used. TA accepts every new configuration which is not much worse than the old one whereas SA accepts worse solutions only with rather small probabilities. An apparent advantage of TA is its greater simplicity. It is not necessary to compute probabilities or to make random decisions.

Differential evolution is a simple and efficient heuristic originally proposed and shown effective for finding global optima for many unconstrained test functions by Storn and Price [20]. Differential evolution is considered one type of evolutionary computational algorithms, which involves the evolution of a population of solutions with size *NP* using operators such as mutation, crossover, and selection. The initial population is often randomly generated following a uniform distribution over the variable domain. A heuristic solution can also be injected as part of the initial population. Each solution vector in the population has to serve once as the target vector so that totally *NP* competitions take place in one generation. For each target vector, the DE's mutation operator generates a new parameter vector, called the mutated vector, by adding the weighted difference between two population-vectors to a third vector in the original version, hence the name of differential evolution. In this study, the version used in [18] is followed, which uses five vectors to generate a mutation vector instead of three. That is,

$$m_i = x_{r1} + F(x_{r2} - x_{r3} + x_{r4} - x_{r5}) \quad (8)$$

where $r1,\ldots,r5$ are mutually different random indexes taking from $\{1, 2, \ldots, NP\}$, and are not equal to i. F in the above equation is a constant real value $\in [0, 2]$, which controls the amplification of the differential variation between the four randomly chosen vectors.

Each mutated vector is subjected to parameter mixing using the crossover operation and the resultant vector is called the trial vector. The trial vector, $\boldsymbol{t_i} = \{t_{i1},\ldots,t_{ij},\ldots,t_{iD}\}$, where D is the dimension

hDE.

Set parameter values (population size *NP* = 250, mutation control factor *F* = 0.6, crossover threshold *CR* = 0.8, the elitism probability p_e = 0.4, tabu list size *tls* = 25, the number of iterations for TA search, *itr4TA* = 1000, and the initial threshold, $thres_0$ = 0.01)
Generate initial solutions of vectors (including both inbound truck sequence and outbound truck sequence) and evaluate their makespans
Determine the overall best solution found so far
Make a duplicate of the current population of vectors
While stopping conditions not met
For each target vector
Pick 5 other vectors different from the target vector
Generate the mutated vector according to Eq. (8)
Generate the trial vector according to Eq. (9)
Convert the trial vector to a sequence by sorting
Evaluate the converted trial vector
Replace the target vector in the duplicate by the converted trial vector if the latter is better
Update the overall best solution if the converted trial vector is better
End
Select the generation best solution not in the tabu list and update the tabu list
Perform threshold accepting based local search on the selected best solution found for *itr4TA* times and update the solution if a trial solution is found better. Set the initial threshold and reduce it as iterations increase.
Make the duplicate as the current population of vectors
If the best solution is not in the current population of vectors, randomly select one to be replaced by the best solution with probability of 0.4.
End

of the problem defined by the number of variables, is obtained from the mutated vector according to

$$t_{ij}=\begin{cases} m_{ij} & \text{if } rand(j)\le CR \text{ or } j=rnbr(i)\\ x_{ij} & \text{if } rand(j)>CR \text{ and } j\ne rnbr(i),\end{cases} \quad (9)$$

where *CR* is the crossover threshold, $rand(j)$ is the jth component of a D-dimensional uniform random number $\in [0, 1]$ and $rnbr(i)$ is a randomly chosen index $\in \{1, \ldots, D\}$ to ensure that at least one mutated dimensional value is used in the trial vector.

If the trial vector yields a lower cost function value than the target vector for a minimization problem, then the trial vector replaces the target vector in the following generation. This constitutes the selection operation. The elitist strategy is adopted with some probability to retain the best solution in the population going to the next generation. The evolution process can be ended with a pre-specified maximum number of generations and/or other criteria such as a pre-specified maximum number of function evaluations or maximal computational time. All vectors in the above-described basic differential evolution algorithm are real and must be converted to a sequence for the subject combinatorial optimization application. Following the random key representation concept, this can be easily achieved by sorting the real values in ascending order and the real values are then replaced by the corresponding positions.

The method used to generate a neighborhood solution in the threshold accepting based local search is the insertion operator, identical to that used in SA and TS. The DE parameters used in Arabani et al. were adopted. Three additional parameters are the tabu list size, *tls*, the number of iterations for TA search, *itr4TA*, and the initial threshold, $thres_0$, for the threshold accepting part. The threshold is decreased as iterations increase in the following manner: $thres = thres_0 \times (1 - itr/itr4TA)$. The values of the three additional parameters *tls*, *itr4TS*, and $thres_0$ are chosen based on trial and error. For readers interested in its implementation, the pseudo code of the hDE algorithm is given below.

Table 1
Test data (entries are estimated numbers of straws).

Fish	20-1	20-2	20-3	30-1	30-2	30-3	40-1	40-2	40-3
1	32	42	50	140	72	56	20	53	82
2	57	54	34	504	60	54	112	24	88
3	504	74	54	56	108	50	50	34	73
4	50	24	74	161	88	24	40	34	72
5	82	54	33	49	54	73	16	47	34
6	56	82	65	53	140	140	112	55	36
7	68	219	60	65	50	65	92	56	56
8	65	53	24	56	56	57	43	16	50
9	108	72	140	60	42	64	47	66	46
10	88	140	247	57	82	61	12	16	65
11	24	64	56	82	83	88	82	147	66
12	112	108	83	92	112	33	23	51	30
13	53	247	68	44	219	60	12	32	38
14	54	83	112	64	200	50	32	247	74
15	56	74	200	83	66	53	22	73	17
16	161	56	82	112	33	108	14	49	11
17	82	56	49	247	44	82	221	31	57
18	44	200	504	112	56	200	34	42	17
19	65	50	92	50	68	92	16	6	112
20	64	88	53	65	112	32	73	41	159
21				54	24	83	113	83	48
22				54	32	34	226	32	20
23				56	64	72	51	504	112
24				24	54	65	82	39	50
25				72	53	82	39	21	40
26				66	65	112	27	161	16
27				112	34	247	65	72	112
28				42	49	58	33	200	92
29				34	61	66	55	26	43
30				60	92	68	219	4	47
31							22	65	12
32							56	20	82
33							59	40	23
34							17	32	12
35							29	49	32
36							64	22	22
37							140	72	14
38							73	36	221
39							27	44	34
40							107	64	16

5. Data and test results

5.1. Material and data collection

5.1.1. Fish sources

Healthy male blue catfish used in this study came from the Baxter Land Company Fish Farm (Arkansas City, Arkansas; 33°34′58.64″N, 91°15′18.45″W). Fish with readily observable secondary sexual characteristics (e.g., a well-muscled head and dark coloration) were selected at the farm. After transport to the Aquaculture Research Station of the Louisiana State University Agricultural Center (Baton Rouge, Louisiana; 30°22′07.32″N, 91°10′27.90″W) in an oxygenated hauler, the fish were maintained in outdoor 0.1-acre ponds and fed commercial diets (Aquaxcel, CargillTM 45% protein) for at least 2 wk. Two days before the start of experiments in April, the fish were captured and moved into indoor tanks with a recirculating system. The bubble-washed bead filters were back-flushed every 2 d. The water quality parameters were: pH 7.0–8.0, ammonia 0.1–0.8 mg/l, nitrite 0.04–0.30 mg/l, alkalinity 39–125 mg/l, hardness 44–126 mg/l, temperature 28 ± 1 °C, and dissolved oxygen 4.3–6.5 mg/l. Cryopreservation experiments were performed at the Aquaculture Research Station.

5.1.2. Data collection

During the years 2009–2011, a total of 104 blue catfish males (17 in 2009, 35 in 2010, and 52 in 2011) were processed using a high-throughput cryopreservation process. Following the steps described in Introduction, sperm were extracted from testes and diluted into HBSS for protection (water activates sperm motility resulting in cell death). The concentrations of sperm suspensions were recorded along with the total volume. The number of straw that can be produced from a single fish, x_i, can be estimated by

$$x_i = \frac{\text{concentration} \times \text{volume}}{5 \times 10^8}. \tag{10}$$

The constant 5×10^8 is the standard number of sperm chosen to be packaged within a straw. The estimated straw numbers for 104 blue catfish males provided the input values used for the grouping problem.

5.2. Test results

The high-throughput cryopreservation system being developed is intended to process 20–30 fish each day. The capacity constraint is limited primarily by the freezer that can process 240 straws in a single freezing cycle. To solve the grouping problem for the current system, 3 data sets of 20 fishes, 3 data sets of 30 fishes, 30 data sets of 40 fishes, and one data set of 84 fishes were randomly selected from the original 104 fishes for testing. Table 1 lists most data sets, except the last one that is given in Appendix A.

The baseline for comparison was established as the current method used for grouping of fishes, i.e., grouping them by the order they were processed. The modified Hoffmann heuristic is deterministic; hence it was run only once for each problem.

For each dataset, each metaheuristic was repeatedly applied 25 times to capture the stochastic nature of the algorithm. Proper

Table 2
Test results of the modified Hoffmann heuristic.

Data	Order	mH	
	f	*f*	Time (s)
20-1	23.9687	7.3474	0.141
20-2	29.2937	16.3517	0.734
20-3	22.4382	6.324	0.078
30-1	26.9601	8.8785	0.656
30-2	16.5294	8.2805	0.437
30-3	21.3027	5.6922	0.375
40-1	20.0707	5.1618	5.515
40-2	16.8081	9.6163	71.625
40-3	23.5250	10.0208	9.531
84-1	24.1183	5.1224	1.0942e5

selection of algorithmic parameter values is important to obtain fairly good results. The optimal set of parameter values is known to be problem dependent. It is difficult and very uncommon to find a common set of parameter values that provides the best result on every problem. However, it is very time consuming to identify the optimal set of parameter values for each problem, especially for combinatorial problems with a large search space such as the one considered in this study. Therefore, a trial-and-error process combined with past experience learned from other studies is often used to select a set of algorithmic parameter values for testing. This common practice was also followed in this study.

All programs were implemented in Matlab. Most datasets and programs were run on a Dell Latitude D830 with Intel Core 2 Duo CPU T930 @ 2.50 GHz. For the 84-1 dataset, the three metaheuristic algorithms were run on a Dell Precision PWS370 with Intel Pentium 4 CPU 3.20 GHz.

5.2.1. Modified Hoffmann heuristic

Table 2 summarizes the test results obtained by the modified Hoffmann (mH) heuristic, along with the baseline results. It can be observed that for each dataset improvement was clearly made by the modified Hoffmann heuristic over the baseline. The effort to optimize the grouping problem is thus well justified. Note that the 84-1 dataset took more than one day to run. As the number of fishes in the full set increases, the size of its power set and the number of subsets for each set in the power set increase as well.

5.2.2. Simulated annealing

The parameter values associated with the simulated annealing algorithm for testing the ten datasets were: $max\_nfe = 30{,}000$, $i_{max} = 500$, $t_{max} = 1000$, and $\alpha = 0.8$. The random selection operator was used to generate trial solutions.

Table 3 summarizes the test results obtained by the simulated annealing algorithm. Note that all results of simulated annealing were better than the baseline. Compared with the results of the modified Hoffmann heuristic, the only case that was better or at least as good in all 25 runs was using the result of the modified Hoffmann heuristic as the initial solution. In this particular case, the 84-1 dataset was not run because it would take too much time. For the remaining 9 datasets, improvement was made on datasets 20-2, 30-2, 40-1, and 40-3. The reason that no improvement was made for the other five datasets (20-1, 20-3, 30-1, 30-3, 40-2) is likely that the solution obtained by the modified Hoffmann heuristic was already optimum. Hence, there was no room for further improvement.

Simulated annealing with the other three initialization methods overall perform worse than the modified Hoffmann heuristic in terms of average function value (specifically, they tie in four data sets and SA win in two and loss in four). However, it should be noted that SA still found equally good or even better solution even when its average function value was worse than that of mH, for most datasets except two (40-1 and 84-1).

5.2.3. Tabu search

The parameter values associated with the tabu search algorithm for testing all 20 problems were: $max\_nfe = 30{,}000$, $nts = 10$, and $tls = 5$. As for the simulated annealing algorithm, the random selection operator is used to generate trial solutions.

Table 4 summarizes the test results obtained by the tabu search algorithm. Note that all results of tabu search were better than the baseline (however, not in all runs when the second initialization method was used on the 40-2 data set). Compared with the results of the modified Hoffmann heuristic, the only case that was better or at least as good in all 25 runs was using the result of the modified Hoffmann heuristic as the initial solution. In this particular case, the 84-1 dataset was not run because it would take too much time. For the remaining 9 datasets, improvement was made on datasets 20-2, 30-2, 40-1, and 40-3; and the improvement amount was larger than that of simulated annealing. The reason that no further improvement was possible for any of the other five datasets is because the optimum was likely attained by the modified Hoffmann heuristic.

Table 3
Test results of simulated annealing.

Data	*f*			CPU time (s)		*f*			CPU time (s)	
	Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-1	1	7.3474	0	10.7537	0	3	7.3474	0	10.7569	0
	2	7.3474	0	10.7194	0	4	7.3474	0	10.8294	0
20-2	1	12.0414	4.963	10.9494	0.0695	3	10.8658	4.963	10.9244	0.0491
	2	13.6087	4.4892	10.95	0.0644	4	11.2577	4.9951	10.9994	0.0657
20-3	1	6.324	0	11.56	0.0402	3	6.324	0	11.5325	0.0427
	2	6.324	0	11.5644	0.0426	4	6.324	0	11.5725	0.0451
30-1	1	8.8785	0	14.6507	0.0395	3	8.8785	0	14.6581	0.0738
	2	8.8785	0	14.6337	0.0435	4	8.8785	0	14.8138	0.0502
30-2	1	11.1937	2.9286	13.415	0.0709	3	11.9767	2.2248	13.3881	0.0681
	2	10.5533	3.3217	13.3919	0.0708	4	7.6317	1.0661	13.5831	0.0836
30-3	1	14.3471	1.8031	13.6844	0.0379	3	13.9865	2.4963	13.6131	0.0601
	2	13.2653	3.3733	13.6306	0.0743	4	5.6922	0	13.8012	0.0334
40-1	1	11.0699	0	20.92	0.6861	3	11.0699	0	22.025	0.5149
	2	10.7844	1.4276	21.7875	0.4825	4	5.1126	0.246	23.4325	0.5312
40-2	1	9.6163	0	19.0375	0.0508	3	9.6163	0	19.0119	0.0705
	2	9.6163	0	18.9475	0.0971	4	9.6163	0	20.9269	0.046
40-3	1	8.3411	0.4733	17.7816	0.5010	3	8.2630	0.4102	20.0095	5.2061
	2	8.4187	0.2516	17.6044	0.2436	4	8.3881	0.249	23.8034	9.317
84-1	1	11.6767	0	77.7656	1.407	3	11.6767	0	77.5192	1.6233
	2	11.6767	0	77.0137	0.1192	4				

Table 4
Test results of tabu search.

Data	f			CPU time (s)		f			CPU time (s)	
	Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-1	1	7.3474	0	11.9675	0.0403	3	7.3474	0	11.9113	0.1058
	2	7.3474	0	11.8731	0.1329	4	7.3474	0	11.8744	0.1586
20-2	1	8.1229	3.6654	11.81	0.1128	3	7.3392	2.7125	11.8768	0.0869
	2	7.731	3.249	11.7381	0.0516	4	8.5147	3.9993	11.83	0.0924
20-3	1	6.324	0	12.7731	0.0672	3	6.324	0	12.7462	0.1113
	2	6.324	0	12.7925	0.072	4	6.324	0	12.7112	0.0413
30-1	1	8.8785	0	16.6432	0.0664	3	8.8785	0	16.6932	0.0835
	2	8.8785	0	16.6844	0.0476	4	8.8785	0	16.8725	0.0535
30-2	1	7.5048	0	14.9832	0.0496	3	8.0898	1.4764	14.9762	0.0551
	2	7.7252	1.2029	14.96	0.0415	4	7.5684	0.5757	15.1444	0.0477
30-3	1	11.4622	4.4167	15.1669	0.0312	3	12.544	3.9298	15.2006	0.0495
	2	11.8228	4.2922	15.2006	0.0461	4	5.6922	0	15.3225	0.0536
40-1	1	11.0699	0	25.7441	10.9862	3	11.0699	0	26.8317	12.9054
	2	11.0699	0	25.7519	10.9544	4	5.1473	0.0726	34.3682	18.04
40-2	1	16.8081	0	21.45	0.3205	3	16.8081	0	20.7375	0.0656
	2	22.5402	3.5594	21.7094	0.41	4	9.6163	0	21.6644	0.0681
40-3	1	6.6541	0.3721	28.3426	16.6371	3	6.7146	0.2273	35.1031	21.2216
	2	6.6067	0.3647	30.3045	17.982	4	6.6782	0.2911	31.9395	18.7559
84-1	1	11.6767	0	93.1541	4.392	3	11.6767	0	90.8041	1.3726
	2	11.6767	0	90.2683	0.9167	4				

Table 5
Test results of ant colony optimization.

Data	f			CPU time (s)		f			CPU time (s)	
	Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-1	1	7.7297	1.9118	87.0644	0.2018	3	7.3474	0	87.0031	0.1509
	2	7.3474	0	87.0875	0.2514	4	7.3474	0	87.4493	0.6794
20-2	1	14.3924	3.9993	83.0425	0.4868	3	14.0006	4.2701	83.4256	0.4489
	2	14.0006	4.2701	81.9919	0.4429	4	15.9598	1.9592	84.1144	2.4286
20-3	1	9.6673	4.5497	91.6737	0.2029	3	15.2395	1.8574	91.9544	0.1451
	2	10.0388	4.6435	92.1925	2.444	4	6.324	0	92.2681	0.1632
30-1	1	9.7617	2.4412	130.9675	0.4232	3	15.65	2.038	131.1481	0.1443
	2	10.6449	3.2083	130.8531	0.2171	4	8.8785	0	131.4275	0.1133
30-2	1	11.8541	1.9673	117.9644	0.1575	3	12.6385	0.9619	117.9218	0.1246
	2	12.2961	1.5206	117.9025	0.2026	4	8.2624	0.0627	118.2306	0.1678
30-3	1	12.9047	3.6806	121.6407	2.5429	3	13.6259	2.9901	123.5881	2.9996
	2	13.9865	2.4963	121.6962	0.1312	4	5.6922	0	126.8125	2.47
40-1	1	11.2128	0.3521	164.5513	2.5114	3	11.1283	0.1993	178.5275	16.6802
	2	11.1597	0.1739	163.9762	0.4094	4	5.1618	0	173.8618	16.7322
40-2	1	10.1916	1.9913	178.5563	2.5352	3	9.6163	0	177.7269	0.5775
	2	9.6163	0	177.8031	0.4697	4	9.6163	0	179.62	0.5385
40-3	1	7.9883	0.3094	162.125	0.5069	3	8.0540	0.3232	161.7569	0.4671
	2	8.0629	0.3525	162.8512	2.709	4	8.0601	0.3886	163.2262	0.4473
84-1	1	16.6233	0.8115	944.7808	7.039	3	16.9479	1.3458	944.883	7.8628
	2	16.9479	1.3458	943.9341	7.6609	4				

Table 6
Test results of hybrid differential evolution.

Data	f			CPU time (s)		f			CPU time (s)	
	Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-1	1	7.3474	0	12.38	0.2046	3	7.3474	0	12.4062	0.2197
	2	7.3474	0	12.3426	0.0821	4	7.3474	0	12.4175	0.0462
20-2	1	6.5555	0	12.7694	0.1202	3	6.5555	0	12.7286	0.2156
	2	6.5555	0	12.7018	0.0788	4	6.5555	0	12.7713	0.0966
20-3	1	6.324	0	13.3144	0.2734	3	6.324	0	13.3176	0.2004
	2	6.324	0	13.395	0.3784	4	6.324	0	13.2494	0.0451
30-1	1	8.8785	0	17.3912	0.3477	3	8.8785	0	17.3744	0.4201
	2	8.8785	0	17.3625	0.3693	4	8.8785	0	17.4018	0.1702
30-2	1	9.0777	2.9699	16.0192	0.1279	3	8.163	2.4576	15.9613	0.0667
	2	9.7912	2.9837	16.0956	0.1809	4	7.2244	0.8459	16.1776	0.1255
30-3	1	7.1347	3.3733	15.0957	0.0312	3	6.0529	1.8031	16.0681	0.0805
	2	6.0529	1.8031	16.0774	0.1600	4	5.6922	0	16.2320	0.143
40-1	1	10.8029	1.3350	18.7012	0.1226	3	10.8029	1.3350	18.7644	0.1402
	2	11.1081	0.1909	18.8063	0.2668	4	5.1618	0	19.7917	0.0696
40-2	1	9.6163	0	19.5013	1.9366	3	9.6163	0	19.0169	0.3266
	2	9.6163	0	19.2108	0.5601	4	9.6163	0	20.9544	0.2727
40-3	1	7.5472	0.7489	17.6193	0.2149	3	7.4025	0.7093	17.5245	0.1364
	2	7.6457	0.8754	17.5512	0.1617	4	7.5122	0.4773	18.8644	0.1173
84-1	1	11.6767	0	77.3268	0.7771	3	11.6767	0	77.2448	0.9906
	2	11.8681	0.9569	77.6144	1.4602	4				

Tabu search with the other three initialization methods overall perform worse than the modified Hoffmann heuristic in terms of average function value (specifically, they tie in three data sets and TS win in three and loss in four). However, it should be noted that for the 30-3 dataset TS still found equally good or even better solution even when its average function value was worse than that of mH.

Comparing the results of tabu search with the results of simulated annealing, it is evident that the tabu search was either equivalent or better regardless of the initialization method for most data sets. The only dataset that TS did worse than SA is the 40-2 dataset.

5.2.4. Ant colony optimization

The values of the parameters associated with ant colony optimization were set as follows: the maximal number of evaluation, $max\_nfe = 30{,}000$, the number of ants, $ants = \lfloor n/5 \rfloor$ (taking the lowest integer), the pheromone evaporation rate, $e = 0.1$, the weight of pheromone, $a = 1$, and the weight of heuristic information, $b = 2$.

Table 5 summarizes the test results obtained by the ant colony optimization algorithm. Note that all results of the ant colony optimization were better than the baseline. Compared with the results of the modified Hoffmann heuristic, the only case that was better was using the result of the modified Hoffmann heuristic as the initial solution. In this particular case, the 84-1 dataset was not run because it would take too much time. For the remaining 9 datasets, improvement was made on two datasets (20-2, 30-2, and 40-3), but the improvement amount was far less than the improvement amounts made by simulated annealing and tabu search. For the 40-1 dataset, unlike SA and TS ACO failed to find any better solution than the mH. The reason that no further improvement was possible for any of the other five datasets was because the optimum was likely attained by the modified Hoffmann heuristic.

Ant colony optimization with the other three initialization methods overall perform worse than the modified Hoffmann heuristic in terms of average function value (specifically, ACO win over mH in two and loss in all others). However, it should be noted that ACO still found equally good or even better solution even when its average function value was worse than that of mH for some datasets except 40-1 and 84-1.

Comparing with the results of tabu search and simulated annealing, it is evident that the ant colony optimization produced worse results, regardless of the initialization method. The CPU time required by ACO (82–131 s) is also longer than that required by SA (11–15 s) and TS (12–17 s). The main reason for this is that ACO employs the sequential construction process in generating a neighborhood solution a single fish at a time.

5.2.5. Hybrid differential evolution

The values of parameters associated with the hybrid differential evolution algorithm were set as follows: population size $NP = 250$, mutation control factor $F = 0.6$, crossover threshold $CR = 0.8$, the elitism probability $p_e = 0.4$, tabu list size $tls = 25$, the number of iterations for TA search, $itr4TA = 1000$, and the initial threshold, $thres_0 = 0.01$.

Table 6 summarizes the test results obtained by the hybrid differential evolution algorithm. Note that all results of hybrid differential evolution were better than the baseline. Compared with the results of the modified Hoffmann heuristic, the only case that was better or at least as good in all 25 runs was using the result of the modified Hoffmann heuristic as the initial solution. In this particular case, the 84-1 dataset was not run because it would take too much time. For the remaining 9 datasets, improvement was made on datasets 20-2, 30-2, and 40-3. Note that for the 40-1 dataset that both SA and TS made slight improvement was not improved by the hybrid differential evolution. The reason that no improvement was made for the other five datasets (20-1, 20-3, 30-1, 30-3, 40-2) is likely that the solution obtained by the modified Hoffmann heuristic was already optimum. Hence, there was no room for further improvement.

Hybrid differential evolution with the other three initialization methods overall perform worse than the modified Hoffmann heuristic in terms of average function value (specifically, they tie

Table 7
Best groupings for datasets 20-1, 20-2, and 20-3.

Dataset	Group	Set	SA	TS	ACO	hDE	f
20-1	1	{3-1}	25	25	25	25	7.3473
	2	{3-2}					
	3	{4,5,9}					
	4	{1,6,10,20}					
	5	{8,14,15,19}					
	6	{2,3-3,11,13,17}					
	7	{7,16}					
	8	{12,18}					
20-2	1	{13-1}	13	20	1	25	6.5555
	2	{1,10,13-2,19}					
	3	{6,9,14}					
	4	{4,11,16,20}					
	5	{2,3,5,8}					
	6	{7}					
	7	{18}					
	8	{12,15,17}					
20-3	1	{10-1}	25	25	25	25	6.324
	2	{18-1}					
	3	{18-2}					
	4	{3,4,14}					
	5	{5,10-2,15}					
	6	{6,12,19}					
	7	{11,16,17,20}					
	8	{1,8,9,18-3}					
	9	{2,7,13}					

Table 8
Best groupings for datasets 30-1, 30-2, and 30-3.

Dataset	Group	Set	SA	TS	ACO	hDE	f
30-1	1	{2-1}	25	25	25	25	8.8785
	2	{2-2}					
	3	{17-1}					
	4	{1,3,13}					
	5	{4,17-2,25}					
	6	{7,12,15}					
	7	{2-3,16,19,21}					
	8	{5,6,8,11}					
	9	{14,22,23,26}					
	10	{10,20,24,29,30}					
	11	{18,27}					
	12	{9,28}					
30-2	1	{13}	2	0	0	0	5.6872
	2	{10,12,27}					
	3	{14,21}					
	4	{1,4,23}					
	5	{5,22,25,30}					
	6	{3,8,15}					
	7	{2,17,18,19}					
	8	{7,11,16,26}					
	9	{6,9,28}					
	10	{20,24,29}					
30-3	1	{27-1}	25	25	25	25	5.6922
	2	{1,23,26}					
	3	{3,6,14}					
	4	{7,19,21}					
	5	{9,16,30}					
	6	{12,18,27-2}					
	7	{22,25,28,29}					
	8	{10,17,20,24}					
	9	{2,5,13,15}					
	10	{4,8,11}					

Table 9
Effect of neighborhood solution generators when SA is used.

Dataset	Gm	*f*			Time (s)		*f*			Time (s)	
		Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-2	1	1	7.3392	2.7125	11.1594	0.0849	3	7.3392	2.7125	11.1537	0.0576
		2	7.3392	2.7125	11.1656	0.0487	4	7.3392	2.7125	11.2131	0.0554
	2	1	6.9473	1.9592	11.7312	0.134	3	7.3392	2.7125	11.8125	0.0691
		2	6.5555	0	11.7869	0.0636	4	6.5555	0	11.9312	0.0509
	3	1	23.4988	0	11.3794	0.0294	3	16.3517	0	10.9306	0.0488
		2	21.1036	4.2551	11.2463	0.255	4	16.3517	0	10.8638	0.0223
	4	1	6.5555	0	11.2138	0.0575	3	6.5555	0	11.2275	0.0627
		2	6.5555	0	11.2081	0.0494	4	6.5555	0	11.3306	0.0405
30-2	1	1	9.1988	3.8218	20.8475	9.5791	3	9.1124	3.6524	21.805	11.3427
		2	8.4848	3.6157	21.9231	11.7241	4	6.8943	1.4792	22.0918	11.7514
	2	1	7.7742	3.2477	24.7969	11.6327	3	10.4692	3.4484	22.0112	9.9516
		2	9.4291	3.5722	23.9544	11.7601	4	7.2789	1.2008	22.17	10.0835
	3	1	14.9664	0	19.8025	8.4689	3	20.724	0	21.9394	11.2955
		2	20.6849	2.9803	21.9807	11.4685	4	8.2806	0	20.9006	11.2256
	4	1	7.8269	2.5987	21.8381	11.4173	3	7.9371	2.7931	21.9556	11.505
		2	7.3187	2.0997	20.2	8.2467	4	6.6942	0.613	22.15	9.5746
30-3	1	1	11.4622	4.4167	16.1375	0.0889	3	10.7409	4.5675	16.0156	0.1193
		2	8.5772	4.2922	16.0406	0.0904	4	5.6922	0	16.3606	0.0884
	2	1	9.6591	4.5675	16.9877	0.1108	3	11.4622	4.4167	16.9958	0.1032
		2	11.8228	4.2922	16.9894	0.1221	4	5.6922	0	17.3342	0.1020
	3	1	21.3027	0	16.454	0.0459	3	21.3027	0	16.4451	0.2090
		2	21.4738	2.0602	16.3594	0.2150	4	5.6922	0	15.0883	0.1018
	4	1	7.4953	3.6806	16.0849	0.0875	3	8.5772	4.2922	16.4479	0.1202
		2	8.2166	4.1314	16.1289	0.1303	4	5.6922	0	16.8431	0.1286

Note: gm = 1, 2, 3, and 4 correspond to pairwise exchange, inverse mutation, shift mutation, and insertion operator, respectively.

in four data sets and hDE win three and loss three). However, it should be noted that for hDE still found equally good or even better solution even when its average function value was worse than that of mH for most datasets, except when the second initialization method was used on the 40-1 dataset.

Tabu search is the best among the previous three algorithms. Comparing to tabu search, the overall performance of the hybrid differential evolution is even better. Specifically, for four datasets, 20-2, 30-3, 40-1, and 40-2, hybrid differential evolution outperforms tabu search while for two datasets, 30-2 and 40-3, the reverse is true.

6. Discussion

For the grouping problem, the sequence that provided the best performance was not unique because the order within each group was irrelevant, so was the order between groups. The best groupings found for the first six datasets are summarized in Tables 7 and 8. In the table, the number of runs made by a metaheuristic algorithm, with the results of the modified heuristic as the initial solution or in the initial population set, to identity the best grouping out of 25 runs is also noted. Note that for dataset 30-2, SA found the lowest *f* value in two runs while TS, ACO, and hDE did not

Table 10
Effect of neighborhood solution generators when TS is used.

Dataset	Gm	*f*			Time (s)		*f*			Time (s)	
		Init	Avg	Stdev	Avg	Stdev	Init	Avg	Stdev	Avg	Stdev
20-2	1	1	7.3392	2.7125	12.2338	0.0335	3	6.9473	1.9592	12.22	0.0292
		2	6.5555	0	12.24	0.0255	4	6.5555	0	12.33	0.1266
	2	1	6.5555	0	12.685	0.0326	3	6.5555	0	12.6169	0.0593
		2	6.9473	1.9592	12.6806	0.0282	4	6.9473	1.9592	12.6469	0.0511
	3	1	23.4988	0	12.2625	0.0248	3	16.3517	0	12.0575	0.1913
		2	22.6566	4.5995	12.2806	0.3615	4	16.3517	0	11.85569	0.1172
	4	1	6.5555	0	12.2262	0.1124	3	6.5555	0	12.1856	0.0613
		2	6.5555	0	12.2194	0.0256	4	6.5555	0	12.225	0.0385
30-2	1	1	11.4234	2.4904	22.24	8.321	3	12.0033	2.1542	24.3156	12.1195
		2	12.1581	1.7482	26.9094	12.6262	4	7.9212	0.5774	22.3256	8.701
	2	1	10.9753	2.7271	25.8637	11.4805	3	11.6581	2.3091	23.9556	10.2984
		2	11.1266	2.4783	23.0069	9.102	4	7.8385	0.6339	28.5719	12.8607
	3	1	14.9664	0	21.7481	8.7939	3	20.724	0	22.85	9.8983
		2	20.6904	2.9198	22.2287	8.4624	4	8.2806	0	22.3169	9.5874
	4	1	7.4836	0.5058	21.9487	8.6433	3	7.43	0.5221	24.0706	10.2004
		2	7.4184	0.5331	21.9832	8.5515	4	7.1815	0.6272	24.6519	10.953
30-3	1	1	10.3803	4.5970	17.6085	0.2312	3	8.9378	4.4167	18.0339	0.3565
		2	8.9378	4.4167	17.7616	0.2156	4	5.6922	0	18.2576	0.2635
	2	1	8.2166	4.1314	18.2953	0.0473	3	7.8560	3.9298	20.0776	3.1503
		2	8.9378	4.4167	18.6094	0.1165	4	5.6922	0	19.0829	2.5859
	3	1	20.3027	0	17.4650	0.0391	3	21.3027	0	17.7494	0.068
		2	20.8534	3.7116	17.3844	0.4115	4	5.6922	0	16.1125	0.0383
	4	1	5.6922	0	17.4675	0.1744	3	6.0529	1.8031	17.4019	0.0799
		2	6.0529	1.8031	17.4138	0.057	4	5.6922	0	17.4881	0.0517

Note: gm = 1, 2, 3, and 4 correspond to pairwise exchange, inverse mutation, shift mutation, and insertion operator, respectively.

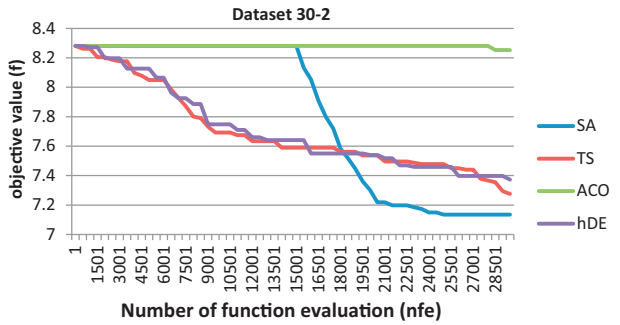

Fig. 2. Average best convergence profiles over 25 runs for all four algorithms.

find it in any run despite the fact that both TS and hDE has slightly lower average *f* value than SA.

Test results obtained by simulated annealing and tabu search, as presented in Sections 5.2.2 and 5.2.3, were based on the random selection operator to generate neighborhood solutions. To investigate whether better results could be obtained by any other operator, three datasets (i.e., 20-2, 30-2, and 30-3), for which SA and TS performed worse than hDE were selected for study. Tables 9 and 10 summarize the test results obtained by the simulated annealing algorithm and the tabu search algorithm using four different neighborhood solution generation operators. The results indicate that for either algorithm the 4th operator, i.e., insertion, was the best. The ranking of the other three operators was generally in the following order: the inverse mutation operator, the pairwise swap, and lastly the shift mutation operator. For datasets 20-2 and 30-3 regardless of whether SA or TS was applied and for dataset 30-2 if SA was applied, the random selection operator was actually the second worst. For dataset 30-2 if TS was applied, the random selection operator had the second best performance, next to the insertion operator only. Therefore, for the subject grouping problem, the insertion operator should be chosen as the preferred operator to be used with simulated annealing and tabu search algorithms. In fact, if the insertion operator is used, then the performances of SA and TS are comparable to that of hDE.

To show how each metaheuristic algorithm converged to the final solution when the solution of the modified heuristic was taken as the initial solution, the average best convergence profile over the 25 runs of SA, TS, ACO, and hDE in solving dataset 30-2 are shown in Fig. 2. The operator used to generate neighborhood solutions for the SA and TS algorithms was the insertion operator. Fig. 2 shows different convergence profiles for the three metaheuristic algorithms as far as this dataset was concerned. Both TS and hDE converged gradually over the course of evaluation. SA and ACO did not improve on the initial solution during the first half of the evaluation. After that, SA improved suddenly and converged quickly while ACO struggled to improve on the initial solution. At the end, SA converged to the lowest value, followed by TS, hDE, and by ACO.

7. Conclusions

To address the grouping problem in a newly developed high-throughput cryopreservation process, this paper has described the mathematical formulation of the problem and also developed one heuristic and four metaheuristic algorithms for its solution. Real world data were collected from a pilot operation. Based on the data collected, datasets of 20, 30, 40, and 84 fishes were generated for testing the performance of the algorithms developed in comparison with the baseline solution of grouping by processing order.

Based on the test results, the following conclusions can be made: (i) improvement over the baseline was substantial if an effort was made to optimize; (ii) the amount of improvement depended upon the algorithm and the initialization method used, the algorithmic parameters and operator selected, and the individual dataset; (iii) among the five neighborhood solution generation operators used in simulated annealing and tabu search, the insertion operator was the best; (iv) all metaheuristic algorithms had the best performance if the result of the modified Hoffmann heuristic algorithm developed was used as the initial solution; and (v) overall the performances of hDE TS, and SA are comparable, if the insertion operator is used in SA and TS, and ant colony optimization has the worse performance.

It was found in this study that use of heuristic methods can significantly improve grouping efficiency. However, compared with the metaheuristic method, heuristic solutions were not as efficient but required less computation time. In practical situations, the efficiency gained from the metaheuristic approach outweighed the time savings of the heuristic methods. A grouping program for blue catfish high-throughput processing has been written based on this study. It can serve as a key organizational step in the process: after sperm concentration adjustment, the operator can use the program for optimal grouping and rearrange the sample order based on the results, before cryoprotectant equilibration. This new step will reduce the operation costs, ensure quality, and accelerate processing. In addition, this program can also be modified and applied for use with high-throughput cryopreservation of biomedical model fishes (e.g., zebrafish, *Danio rerio*). There is an estimated backlog of millions of samples for these animals – at least 20,000 research lines need to be cryopreserved and banked in germplasm repositories (www.zfin.org). No true high-throughput process currently exists for these species.

Possible topics for future study include: (i) improving the performance of the ACO algorithm by developing better heuristic information to be used with the pheromone information; (ii) tailoring other metaheuristic algorithms for the subject application to investigate their performance; (iii) considering uncertain characteristics of the process, e.g., capacity constraint not as being exact, but as interval or fuzzy values; and (iv) solving the problem as a multiple-objective case, rather than as a weighted-average single objective case.

Acknowledgements

We thank A. Saale, C. Staudermann, D. Kuenz, M. Doan, N. Novelo, H. Pizzitola, R. Uribe, for technical assistance during the spawning season, data collection, and suggestions. We thank Baxter Land Company for providing blue catfish males. This work was supported in part by funding from the Louisiana Sea Grant College Program, USDA special grants, USDA-SBIR, and the National Center for Research Resources of the National Institutes of Health (R24RR023998). This manuscript was approved by the director of the Louisiana Agricultural Experiment Station as number 2011-244-6258.

Appendix A.

The 84-1 dataset

20, 112, 50, 40, 16, 112, 92, 43, 47, 12, 82, 23, 12, 32, 22, 14, 221, 34, 16, 73, 113, 226, 51, 82, 39, 27, 65, 33, 55, 219, 22, 56, 59, 17, 29, 64, 140, 73, 27, 107, 64, 49, 24, 53, 24, 34, 34, 47, 55, 56, 16, 66, 16, 147, 51, 32, 247, 73, 49, 31, 42, 6, 41, 83, 32, 504, 39, 21, 161, 72, 200, 26, 4, 65, 20, 40, 32, 49, 22, 72, 36, 44, 64, 22.

References

[1] P. Mazur, S.P. Leibo, G.E. Seidel Jr., Cryopreservation of the germplasm of animals used in biological and medical research: importance, impact, status, and future directions, Biology of Reproduction 78 (1) (2008) 2–12.

[2] C. Polge, A.U. Smith, A.S. Parkes, Revival of spermatozoa after vitrification and sehydration at low temperatures, Nature 164 (4172) (1949) 666.
[3] C. Polge, L.E.A. Rowson, Fertilizing capacity of bull spermatozoa after freezing at −79 °C, Nature 169 (4302) (1952) 626–627.
[4] M. Thibier, H.G. Wagner, World statistics for artificial insemination in cattle, Livestock Production Science 74 (2) (2002) 203–212.
[5] J.H.S. Blaxter, Sperm storage and cross-fertilization of spring and autumn spawning herring, Nature 172 (4391) (1953) 1189–1190.
[6] T.R. Tiersch, P.M. Mazik, Cryopreservation in Aquatic Species, The World Aquaculture Society, Baton Rouge, LA, 2000.
[7] USDA-NASS, 2005 Census of Aquaculture, USDA National Agricultural Statistics Service, 2005.
[8] M. Masser, R. Dunham, Production of Hybrid Catfish, Southern Regional Aquaculture Center (SRAC), SRAC 190, 1998.
[9] B.P. Boever, Analysis of U.S. aquacultural producer preferences for genetic improvement and cryopreservation, in: Agricultural Economics & Agribusiness, Louisiana State University. Master of Science (M.S.), Baton Rouge, LA, 2006.
[10] A.N. Bart, D.F. Wolfe, R.A. Dunham, Cryopreservation of blue catfish spermatozoa and subsequent fertilization of channel catfish eggs, Transactions of the American Fisheries Society 127 (5) (1998) 819–824.
[11] P.R. Lang, K.L. Riley, J.E. Chandler, T.R. Tiersch, The use of dairy protocols for sperm cryopreservation of blue catfish (*Ictalurus furcatus*), Journal of the World Aquaculture Society 34 (1) (2003) 66–75.
[12] E Hu, H. Yang, T.R. Tiersch, High-throughput cryopreservation of spermatozoa of blue catfish (*Ictalurus furcatus*): establishment of an approach for commercial-scale processing, Cryobiology 62 (1) (2011) 74–82.
[13] T.R. Hoffmann, Assembly line balancing with a precedence matrix, Management Science 9 (4) (1963) 551–562.
[14] S.G. Pennambalam, P. Aravindan, G.M. Naidu, A comparative evaluation of assembly line balancing heuristics, International Jounal of Advanced Manufacturing Technology 15 (1999) 577–586.
[15] S. Kirpatrick, C.D. Gelatt Jr., C.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.
[16] F. Glover, Tabu search – a tutorial, Interfaces 20 (4) (1990) 74–94.
[17] M. Dorigo, Optimization, Learning, and Natural Algorithms, Politecnico di Milano, Italy, 1992.
[18] A.B. Arabani, S.M.T.F. Ghomi, M. Zandieh, Metaheuristics implementation for scheduling of trucks in a cross-docking system with temporary storage, Expert Systems with Applications 38 (2011) 1964–1979.
[19] G. Dueck, T. Scheuer, Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing, Journal of Computational Physics 90 (1990) 161–175.
[20] R. Storn, K. Price, Different evolution – a simple and effective heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (1997) 341–359.