Ann Oper Res (2010) 180: 63–103
DOI 10.1007/s10479-008-0493-0

Multi-objective and prioritized berth allocation in container ports

C.Y. Cheong · K.C. Tan · D.K. Liu · C.J. Lin

Published online: 13 December 2008

Abstract This paper considers a berth allocation problem (BAP) which requires the determination of exact berthing times and positions of incoming ships in a container port. The problem is solved by optimizing the berth schedule so as to minimize concurrently the three objectives of makespan, waiting time, and degree of deviation from a predetermined priority schedule. These objectives represent the interests of both port and ship operators. Unlike most existing approaches in the literature which are single-objective-based, a multi-objective evolutionary algorithm (MOEA) that incorporates the concept of Pareto optimality is proposed for solving the multi-objective BAP. The MOEA is equipped with three primary features which are specifically designed to target the optimization of the three objectives. The features include a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure. The effects that each of these features has on the quality of berth schedules are studied.

Keywords Berth allocation problem · Evolutionary algorithms · Multi-objective optimization · Combinatorial problems

1 Introduction

In many container ports in the US and Japan, berths are leased directly to ship operators where they have exclusive use of the berths. These ports are known as dedicated terminals. The ship operators themselves are in charge of running the operations of the berths. For a ship operator handling large volume of containers and ship calls, the productivity will be high due to economies of scale. However, overcapitalization of the port might result

C.Y. Cheong · K.C. Tan (✉) · C.J. Lin
Department of Electrical and Computer Engineering, National University of Singapore, 4 Engineering Drive 3, Singapore 117576, Singapore
e-mail: eletankc@nus.edu.sg

D.K. Liu
ARC Centre of Excellence for Autonomous Systems (CAS), Faculty of Engineering, University of Technology, PO Box 123, Broadway, Sydney, NSW 2007, Australia

if the handled volume is small as operations will be costly (Imai et al. 2001). Multi-user terminals, commonly found in Europe and East Asia, on the other hand are completely run by port operators who will assign incoming ships to any berth, not necessarily the same berth, whenever they call at the port. This type of ports is especially popular in land-scarce countries, such as Singapore and Hong Kong, as they have limited land that can be set aside for berths. The productivity of these ports depends largely on the efficient berth allocation of calling vessels. From the point of view of ship operators, timeliness is an important factor as delays at one port often result in a cascading effect of late port calls at subsequent ports of call for the ship. This can result in heavy losses for shipping companies. Thus, the effective allocation of berths to ships is indeed a very complex and challenging issue that is of concern to both port operators and shipping companies.

This paper considers the berth allocation problem (BAP) in multi-user terminals. Given a collection of ships that are to arrive at the port within a planning horizon, the BAP involves determining the berthing time and location of each of the ships, while satisfying a number of spatial and temporal constraints, to optimize operations. More details of the problem are given in Sect. 2.3.

A number of objectives for optimizing port throughput have been considered in the literature. Imai et al. (2001, 2003, 2005, 2007) considered the BAP by minimizing the total service time of ships. The service time of each ship includes the waiting time between the arrival time of the ship at the port and the time the ship berths as well as the handling time for loading or unloading of containers. Guan et al. (2002) developed a heuristic for the BAP with the objective of minimizing the total weighted completion time of ship services. Kim and Moon (2003) solved their version of the BAP by minimizing the penalty cost resulting from delays in the departure of ships and additional handling costs resulting from non-optimal locations of ships in the port. According to their formulation, each ship has an optimal berthing location in the port. Park and Kim (2003) solved the same problem by using a sub-gradient method. In their formulation, additional cost is incurred from early or late start of ship handling against their estimated time of arrivals. Li et al. (1998) solved the BAP by minimizing the makespan of the schedule. Imai et al. (2003) tackled the BAP with service priority, where some ships are given priority, in terms of being serviced earlier, over others. In their work, they provided several examples and arguments for differentiating the service treatment of ships. Lim (1998) took a different approach to the BAP by minimizing the maximum amount of space used for berthing ships. Lai and Shih (1992) proposed some heuristic algorithms for a BAP which is motivated by the need for more efficient berth usage at the HIT terminal of Hong Kong. Their problem assumes the first-come-first-serve (FCFS) allocation strategy which in most cases does not lead to optimal schedules. Imai et al. (1997) considered a BAP for commercial ports. Most service queues are traditionally processed on a FCFS basis. They concluded that for high port throughput, optimal ship-to-berth assignments should be found without considering the FCFS heuristic. However, they also noted that this may result in some dissatisfaction among ship operators regarding the order of the service sequence.

Given the many objectives that have been used to formulate the BAP, it is rather surprising that very little work has been done in the area of multi-objective optimization in BAPs. From the studies of Imai et al. (1997), it is apparent that the BAP is inherently a multi-objective optimization problem. An ideal berthing plan for ship operators is one where ships do not have to wait to be berthed and be serviced in the shortest possible time. However, an ideal berthing plan for port operators is one where the makespan, i.e. the time between the first ship that berths at the port and the last ship that leaves, is minimal to achieve full use of their resources at all times. Thus, as port operators try to achieve high throughput in their ports, the satisfaction of ship operators should be considered concurrently. Despite this, most of the existing literature uses single-objective-based heuristic methods that incorporate penalty functions or combine the different objectives by a weighting function (Imai et al. 1997, 2003). The drawback of such an objective function approach is that the weights are difficult to be determined precisely, especially when there is insufficient information or knowledge concerning the large real-world BAP. Clearly, these issues can be easily addressed by taking a multi-objective approach that optimizes all objectives concurrently and effectively without the need of calibrating weighting coefficients.

In this paper, a multi-objective evolutionary algorithm (MOEA) (Tan et al. 2007; Cheong et al. 2007) is applied to minimize multiple conflicting objectives from the points of view of port and ship operators. In contrast to existing single-objective-based approaches, it utilizes the concepts of Pareto optimality to minimize concurrently the makespan of the port and the dissatisfaction of ship operators by reducing the waiting times of their ships. In addition, the MOEA is designed to handle service priority by including the degree of adherence to a predetermined priority schedule as a third objective. This is to allow the port flexibility in giving service priority to ships. Reasons for maintaining a priority system could include terms laid down in shipping contracts, affluence of shipping companies, preference of handling ships with larger or smaller container volume first, or simply the preference of the port management to adopt a FCFS policy to avoid complaints from shipping companies of unfair treatment. To solve this multi-objective optimization problem, the MOEA is equipped with three primary features which are specifically designed to target the optimization of the three objectives. The features include a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure. The effects that each of these features has on the quality of berth schedules will be studied.

This paper is organized as follows. Section 2 gives some background information on multi-objective optimization and evolutionary algorithms as well as the problem formulation of the BAP that is being considered in this paper. Section 3 presents the program flow and features of the proposed MOEA. Section 4 presents extensive simulation results and analysis of the proposed algorithm. Conclusions are drawn in Sect. 5.

2 Background information

2.1 Multi-objective optimization

As mentioned in the introduction, the BAP is a multi-objective optimization problem where a number of objectives, from the points of view of both port and ship operators, need to be optimized concurrently. These objectives can be non-commensurable and conflicting in nature (Imai et al. 1997). In contrast to single-objective optimization, the solution to a multi-objective optimization problem exists in the form of alternate tradeoffs known as the Pareto optimal set. Each objective component of any non-dominated solution in the Pareto optimal set can only be improved by degrading at least one of its other objective components. In the total absence of information regarding the preference of objectives, a ranking scheme based upon the Pareto optimality is regarded as an appropriate approach to represent the fitness of each solution in an evolutionary algorithm for multi-objective optimization.

Thus, the role of multi-objective optimization in the BAP is to discover such a set of Pareto-optimal solutions from which the decision maker can select an optimal solution based on the situation at hand. The Pareto fitness ranking scheme (Fonseca 1995) for evolutionary multi-objective optimization is adopted in this paper to assign the relative strength of

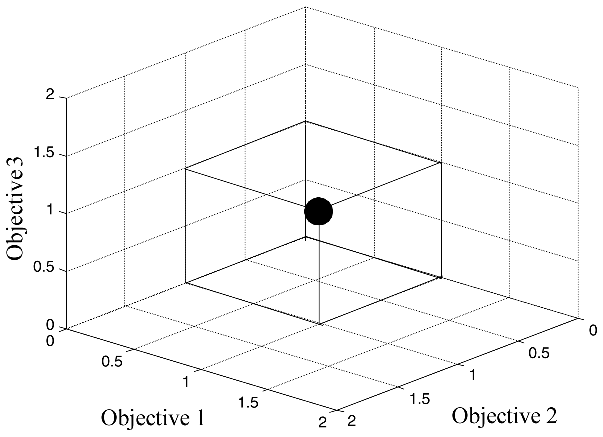


Fig. 1 Example to show how a solution is dominated by another solution

solutions. The ranking approach assigns the same smallest rank for all non-dominated solutions, while the dominated ones are inversely ranked according to the number of solutions dominating them. In Fig. 1, a hypothetical solution (the black dot) is plotted in the objective domain at coordinates (1, 1, 1). The problem is assumed to involve the minimization of the three objectives. Each solution will define a rectangular box encompassing the origin (hypothetical ideal point) as shown in the figure. Another solution will dominate this solution if and only if it is within or on the box defined by the first solution but not equal, in terms of all the three objectives, to the first solution.

2.2 Evolutionary algorithms

In the search for the Pareto optimal set for the BAP, this paper relies on the optimization capabilities of evolutionary algorithms. Evolutionary algorithms are a general purpose optimization tool inspired by Darwin's theory of evolution. They have the capability to produce optimal or near-optimal solutions for multi-dimensional problems and thus have been successfully applied to a wide variety of problems (Ross and Corne 1994).

Evolutionary algorithms operate on a population of solutions represented by some coding, mapping each solution onto a chromosome. Each chromosome consists of a number of genes, each representing a unit of information. The search for new solutions is effected by combining genes from different chromosomes of the population (crossover) to produce offspring or by altering existing chromosomes of the population (mutation). A simulation of 'natural selection' then takes place by first evaluating the quality of each chromosome and then selecting the fittest ones to survive in the next generation. The surviving chromosomes at the end of the search are then decoded into the corresponding solutions.

Evolutionary algorithms have been applied by Ying (1995), Foo (2000), Nishimura et al. (2001), and Imai et al. (2003, 2007) on the BAP but all of them took a single-objective approach towards the problem.

2.3 Problem formulation

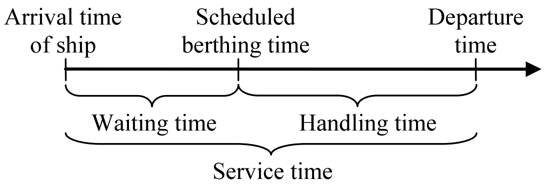
The BAP involves allocating a fixed number of berths to a number of ships arriving at the port within the planning horizon for container handling by determining the berthing time and location of each ship. In essence, the BAP bears some resemblance to machine scheduling problems (Li et al. 1998; Guan et al. 2002), with berths analogous to machines and ships analogous to tasks. However, there are also a number of constraints that are exclusive to the BAP and set it apart from machine scheduling problems.

There are generally two types of berth allocation schemes in the literature. Discrete BAP (Lai and Shih 1992; Brown et al. 1994, 1997; Imai et al. 1997, 2001, 2003) considers the berthing space to be a collection of discrete berthing sections where each ship, in terms of length, must fit within the perimeter of its allocated section and only one ship can be serviced in each section at any time. Such a scheme simplifies the problem since it only requires the solver to allocate ships to the finite number of discrete sections. On the other hand, continuous BAP (Li et al. 1998; Lim 1998; Guan et al. 2002; Park and Kim 2002, 2003; Kim and Moon 2003) considers the berthing space to be a continuous stretch and multiple ships can be berthed simultaneously along the stretch as long as the ships are within the perimeter of the space. This scheme is more complex as it requires the solver to determine the exact berthing location of each ship along the continuous stretch but it allows the berthing space to be utilized more efficiently. Most of the existing literature that adopted the continuous berth allocation scheme limited their studies to a single continuous berthing stretch where all ships are scheduled within the stretch (Li et al. 1998; Lim 1998; Guan et al. 2002; Park and Kim 2002, 2003; Kim and Moon 2003). This paper adopts a more general scheme where the entire berthing space consists of a number of discrete sections and the space is continuous within each section. Each of these discrete sections represents a berth. This scheme is a hybridization of the discrete and continuous BAPs in that the problem involves the allocation of incoming ships to berths and the determination of the exact berthing location of each ship within its allocated berth. Such a scheme is closer to real-world settings where a port consists of a number of berths which are separated geographically. This hybrid scheme has been adopted by Nishimura et al. (2001) but their formulation does not require the determination of the exact berthing location of each ship within its allocated berth. In their work, ships are allowed to be serviced simultaneously as long as the sum of their lengths does not exceed the berth length. In reality, this is often not the case. If a ship occupies the centre of a berth leaving empty berth space at the sides, the succeeding ship may not be able to berth within the perimeter of the berth even if the berth length condition is satisfied. Therefore, a more relevant BAP is one where the exact berthing positions of ships are determined. On top of the physical constraint that a ship must be berthed within the perimeter of its allocated berth, each berth also has a water depth and ships with drafts larger than the depth are not allowed to be allocated to the berth.

Regardless of berth allocation scheme, the sequence of events that takes place for each ship calling at the port is the same. Each of the ships will come into the port, wait for the scheduled berthing time, berth at the designated position within the allocated berth, load or unload containers, and leave the port. Figure 2 shows the berth operation timeline for each ship. In Fig. 2, the service time of a ship at the port includes the waiting time between the arrival time of the ship and the time the ship berths as well as the handling time for loading or unloading containers.

The handling time for each ship is different at different berths. This is to take into account the transportation time for moving the containers to be loaded onto the ship from the original

Fig. 2 Berth operation timeline



storage area to the allocated berth (Nishimura et al. 2001; Imai et al. 2001, 2003, 2005). This handling time is also assumed to be deterministic.

The BAP studied in this paper is then formulated as follows:

Minimize

$$\operatorname*{Max}_{i\in V}(d_i) - \operatorname*{Min}_{i\in V}(b_i) \tag{1}$$

$$\sum_{i\in V}(b_i - a_i) \quad \text{and} \tag{2}$$

$$\sum_{i\in V}\frac{(bo_i - po_i) + |bo_i - po_i|}{2} \tag{3}$$

subject to

$$\sum_{j\in B} x_{ij} = 1 \qquad \forall i \in V \tag{4}$$

$$b_i - a_i \geq 0 \qquad \forall i \in V \tag{5}$$

$$(DB_j - DS_i)x_{ij} \geq 0 \qquad \forall i \in V,\ \forall j \in B \tag{6}$$

$$(LB_j - LS_i)x_{ij} \geq 0 \qquad \forall i \in V,\ \forall j \in B \tag{7}$$

$$\sum_{j\in B}\left(LB_j - \sum_{i'\in V-\{i\}} LS_{i'} y_{ii'} x_{i'j} - LB_j\right)x_{ij} \geq 0 \qquad \forall i \in V \tag{8}$$

$$|p_{ij} - p_{i'j}|\delta^p_{ii'j} \geq \frac{LS_i - LS_{i'}}{2}\delta^p_{ii'j} \qquad \forall i, i' \in V\ (i' \neq i),\ \forall j \in B \tag{9}$$

$$\left|\frac{b_i + d_i}{2} - \frac{b_{i'} + d_{i'}}{2}\right|\delta^t_{ii'}j \geq \frac{h_{ij} + h_{ij'}}{2}\delta^t_{ii'j} \qquad \forall i, i' \in V\ (i' \neq i),\ \forall j \in B \tag{10}$$

$$\delta^p_{ii'j} + \delta^t_{ii'j} = 1 \qquad \forall i, i' \in V\ (i' \neq i),\ \forall j \in B \tag{11}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V,\ j \in B \tag{12}$$

$$p_{ij} \in \mathbb{Z}^+ \qquad \forall i \in V,\ j \in B \tag{13}$$

$$b_i \in \mathbb{Z}^+ \qquad \forall i \in V \tag{14}$$

where B is the set of berths, V is the set of ships, a_i is the arrival time of ship i, b_i is the berthing time of ship i, d_i is the departure time of ship i, h_{ij} is the handling time of ship i at berth j, po_i is the priority order of ship i, bo_i is the berthing order of ship i, DB_j is the water depth of berth j, DS_i is the draft of ship i including the safety vertical distance for berthing, LB_j is the length of berth j, LS_i is the length of ship i including the safety horizontal length, and p_{ij} is the position of ship i in berth j. $x_{ij} = 1$ if ship i is serviced at berth j, $x_{ij} = 0$ otherwise. $y_{ii'} = 1$ if ship i begins its service when ship i' is being serviced at the same berth, $y_{ii'} = 0$ otherwise. $\delta^p_{ii'j} = 1$ if the non-overlapping restriction of berth space in berth j is applied for ships i and i', $\delta^p_{ii'j} = 0$ otherwise. $\delta^t_{ii'j} = 1$ if the non-overlapping restriction of time in berth j is applied for ships i and i', $\delta^t_{ii'j} = 0$ otherwise.

In the problem formulation above, function (1) represents the objective of minimizing the makespan of the port. The makespan is defined as the amount of time between the first ship that berths and the last ship that leaves the port. The second objective is represented by function (2), which minimizes the total waiting time incurred by ships. Function (3) represents the third objective of adhering as closely as possible to a predetermined priority schedule by minimizing the total number of crossings between ships. The berthing order is derived by arranging the scheduled ships, regardless of berth, in order of increasing berthing times. The first ship that berths is given a berthing order value of 1, while the second ship is given a value of 2 and so on. Similarly, the ship that is given the highest berthing priority is assigned a priority order value of 1, while the next ship is given a value of 2 and so on. The number of crossings contributed by a particular ship is then defined as the difference between its berthing order and priority order when its berthing order is greater than its priority order ($bo_i > po_i$). These three objectives constitute the multi-objective nature of the BAP considered in this paper. Constraint (4) ensures that every ship can only be serviced at one berth without disruption. Constraint (5) ensures that ships are serviced only after their arrivals. Constraints (6) and (7) guarantee that the berths that ships are allocated satisfy the physical properties in terms of berth length and water depth. Constraint (8) ensures that the sum of the lengths of ships being serviced simultaneously at a berth does not exceed the length of the berth. Constraints (9) and (10) are the non-overlapping restrictions. Constraint (11) requires that for ships berthed at the same berth, either non-overlapping in berth space or time should be satisfied at all times, i.e. ships allocated to the same berth are not allowed to overlap in terms of both space and time as that would signify a collision. Constraints (12), (13), and (14) show the domains of the three decision variables of the BAP.

3 Multi-objective evolutionary algorithm

Having formulated the BAP as a multi-objective problem, this section presents the multi-objective evolutionary algorithm (MOEA) proposed to solve the BAP by minimizing concurrently the three objectives of makespan of the port, waiting times of ships, and number of crossings. The main features of the MOEA will first be introduced in turn before describing the algorithmic flow.

3.1 Fixed-length chromosome

A fixed-length chromosome representation (Fig. 3) is used in the MOEA. Each chromosome encodes a complete and feasible berth schedule and consists of a fixed number of berths. Each berth consists of a number of ships that are allocated to the berth. The order of ships within each berth indicates the order in which the ships are assigned berthing space and

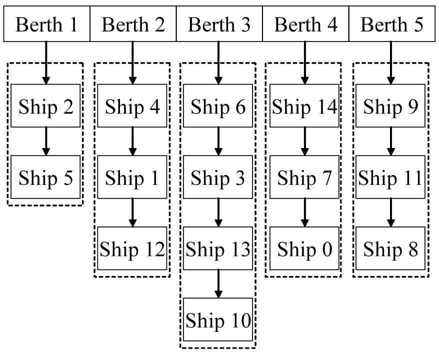


Fig. 3 Fixed-length chromosome representation

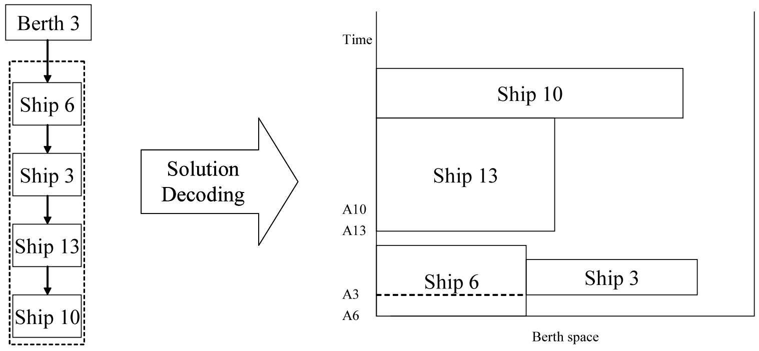


Fig. 4 Illustration of solution decoding

time. This assignment is carried out using two different solution decoding schemes which will be described in the next section. In the figure, ships 2 and 5 are allocated to berth 1 and ship 2 is assigned berthing space and time before ship 5.

3.2 Solution decoding

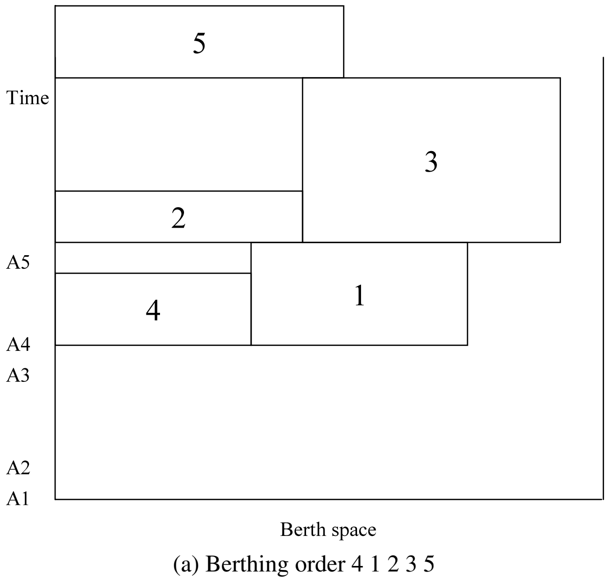
Given the order of ships in each of the berths in a chromosome, the MOEA has to decode the candidate solution by assigning the exact berthing positions and times of the ships (Fig. 4). In this way, the departure, waiting, and service times of the ships can be determined, leading to the fitness or objective values of the chromosome.

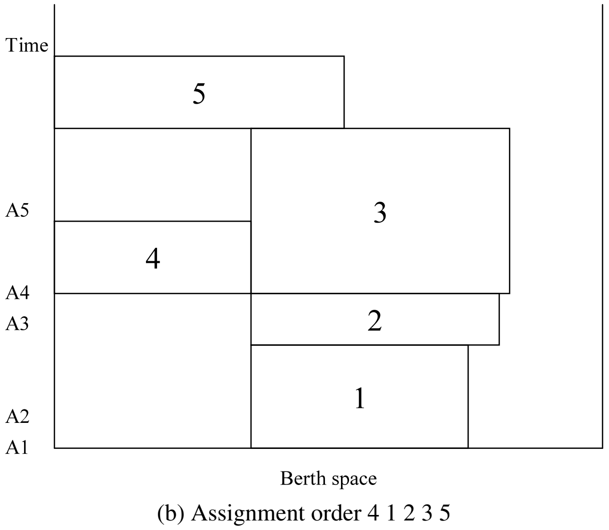
From Fig. 4, it can be seen that the schedule of a particular berth in the port can be represented by a two-dimensional plane. The horizontal axis represents the position in the berth, while the vertical axis is the time axis. Each ship is represented by a rectangle such that the length of the rectangle is the length of the ship and the height of the rectangle is the handling time of the ship at the berth. The bottom-left corner of the rectangle represents the berthing time of the ship while the top-left corner represents its departure time. A6, A3, A13, and A10 in the figure represent the arrival times of the respective ships.

A simple solution decoding scheme to obtain the berth schedule is to treat the order of ships within each berth of a chromosome as the berthing order, i.e. a ship can only berth at the same time or later than its preceding ship. This scheme will be referred to as the berthing order decoding scheme. An example of how this decoding scheme works is illustrated in Fig. 5(a). The order of ships for the particular berth in the chromosome is ship 4 $\rightarrow$ ship 1 $\rightarrow$ ship 2 $\rightarrow$ ship 3 $\rightarrow$ ship 5. In Fig. 5(a), ship 4 is first assigned the leftmost position of the berth as soon as it arrives at A4. Next, due to the solution decoding scheme, ship 1 cannot berth at A1 even though berth space is available. It can only berth at the same time or later than ship 4. Since the berth is long enough to accommodate the simultaneous servicing of ships 1 and 4, ship 1 berths at A4 next to ship 4. On the other hand, ship 2 cannot berth at A4 as the berth is not long enough to accommodate the simultaneous servicing of the three ships. The departure of ship 4 also does not release enough berth space to accommodate ship 2. As such, ship 2 is assigned the leftmost position of the berth after ship 1 has departed from the berth. At the same time, ship 3 berths alongside ship 2 since it has already arrived at A3. Lastly, ship 5 has to wait until ship 3 has left the berth before it gets to berth due to insufficient berth space.

Other than the berthing order decoding scheme, this paper proposes another decoding scheme which treats the order of ships within each berth of a chromosome as the assignment

Fig. 5 Illustration of different solution decoding schemes





order, assigning a ship a feasible berth space with the earliest possible berthing time starting from the left. In this scheme, a ship may berth earlier than its preceding ships as long as it has arrived at the port and berth space is available. This scheme will be referred to as the assignment order decoding scheme. An example to show how this decoding scheme works is illustrated in Fig. 5(b). The order of ships in the chromosome is the same as that used in the berthing order decoding scheme example. Like the berthing order decoding scheme, ship 4 is assigned the leftmost position of the berth as soon as it arrives at A4. A characteristic of this decoding scheme is that the assignment of berth space to a particular ship renders the berth space unavailable to succeeding ships until the ship has left the berth, i.e. in this case the berth space occupied by ship 4 is made unavailable to ships 1, 2, 3, and 5 until ship 4 has departed from the berth. Unlike the berthing order decoding scheme, ship 1 is allowed to berth at A1. However, the scheme dictates that ships are always assigned the leftmost position of the berth whenever it is available. Since the space to be occupied by ship 4 has already been rendered unavailable, ship 1 berths at the earliest leftmost available berth space, which is the space next to ship 4. Another characteristic of this scheme is that it has two main criteria for determining the berthing location of each ship—earliest possible berthing time and leftmost position, with the former taking precedence over the latter. It is for this reason that ship 1 is not berthed at a position where it has to wait for ship 4 to complete servicing even though that position is to the left of its assigned berthing position. Next, ship 2 is unable to berth beside ship 1 at A2 since the available berth space is not long enough to accommodate the ship. As such, it takes over the berth space from ship 1 after ship 1 has departed from the berth. The same reason explains the assignment of berth space and time to ship 3. With the inclusion of ship 3 in the schedule, the available berth space includes the space previously occupied by ship 4 and the space next to ship 3. However, both spaces are not long enough to accommodate ship 5. As such, ship 5 can only berth after ship 3 has left the berth and it berths at the leftmost position of the berth.

From Fig. 5, it can be seen that the berth schedules obtained using the two decoding schemes are very different even though they originated from the same chromosome. This implies that a chromosome may have two different sets of objective values based on the two decoding schemes. The effects of the two solution decoding schemes on berth schedule quality will be studied and discussed in Sect. 4.2.

As a sidenote, one can observe that the berth schedules in Fig. 5 are not favorable as there are voids in the berth schedules resulting in inefficient usage of the berth space. Figure 6 shows the berth schedule decoded from the sequence ship 1 → ship 2 → ship 3 → ship 4 → ship 5. In this case, both solution decoding schemes lead to the same berth schedule. It is obvious that the schedule in Fig. 6 has a lower makespan and waiting time than the two schedules in Fig. 5. From Figs. 5 and 6, it is clear that the order of ships in a chromosome is an important consideration as it will affect how the berth schedule turns out. Inefficient berth schedules will result in unsatisfactory objective values of makespan, waiting time, and number of crossings.

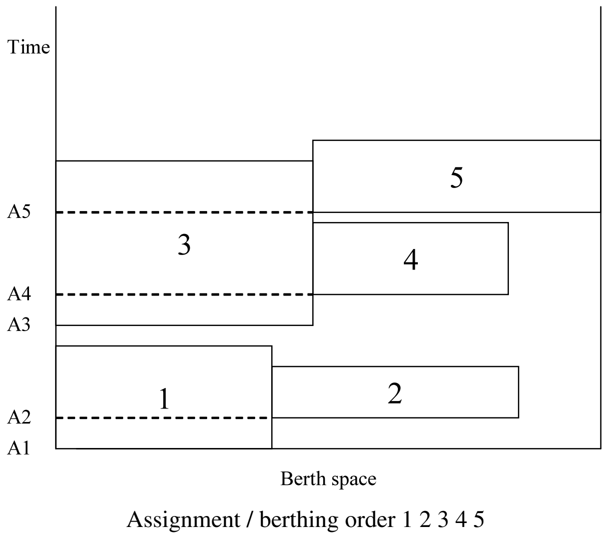


Fig. 6 A more favorable berth schedule

3.3 Berth-exchange crossover

Berth-exchange crossover allows sequences of genes in a fit chromosome to be shared with other chromosomes in the evolving population. It involves the exchange of berths between pairs of parent chromosomes, chosen based on the crossover rate, to produce offspring chro-

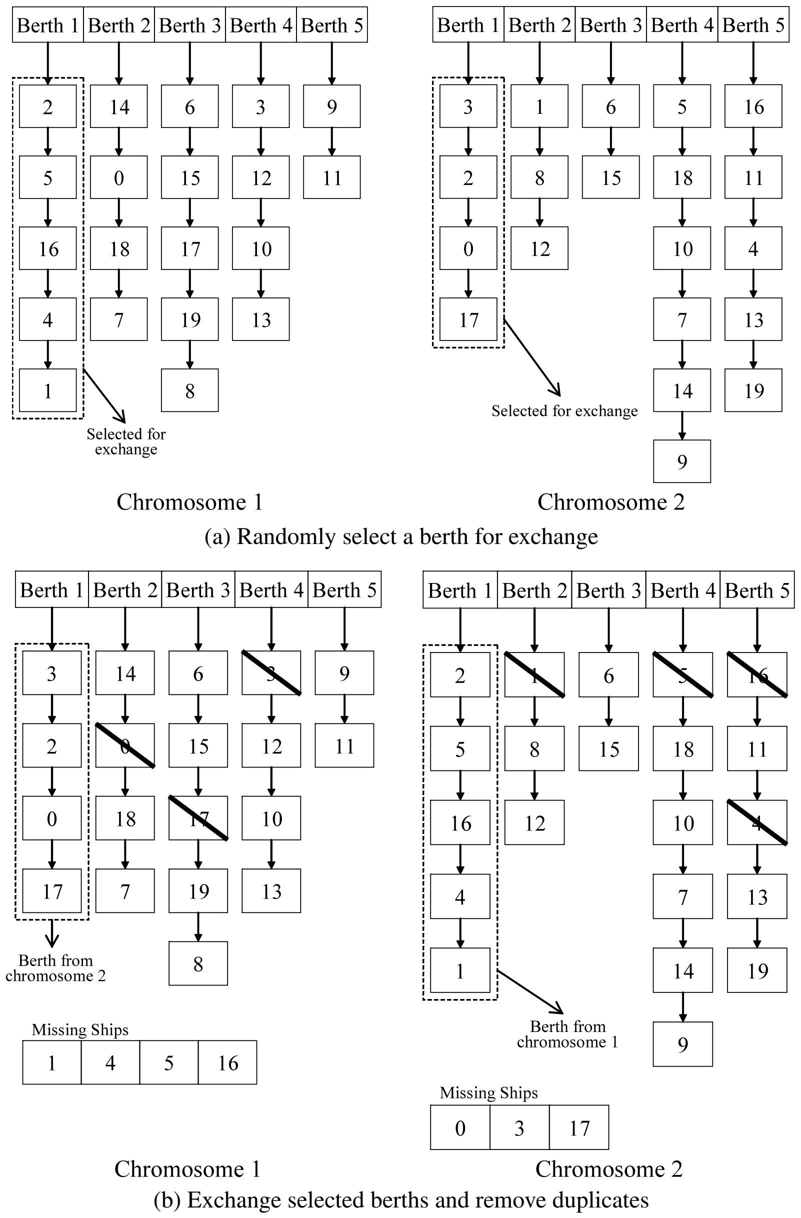
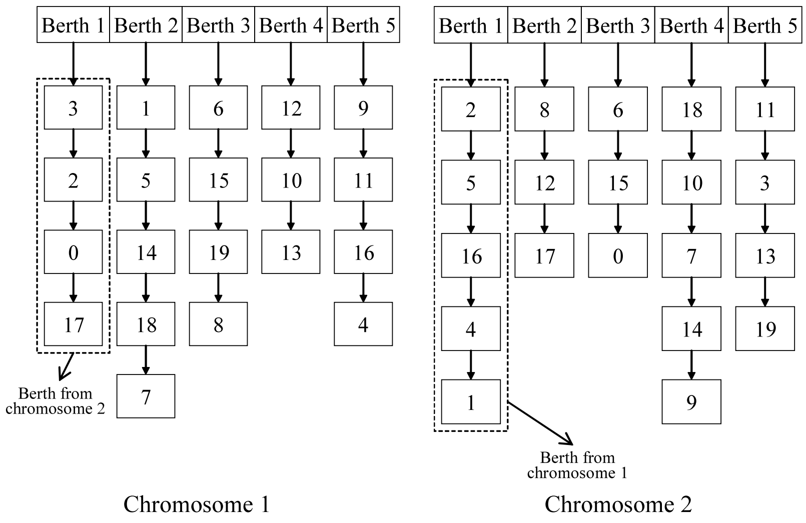


Fig. 7 Illustration of berth-exchange crossover



(c) Reinsertion of missing ships to form offspring

Fig. 7 (*Continued*)

mosomes. The operation of berth-exchange crossover is shown in Fig. 7. The berth to be exchanged is selected at random and applies to both the parent chromosomes. As the two identically indexed berths from each pair of parents have the same berth length and water depth, the exchange of the list of ships at the berth will not result in ships being allocated to a berth where they do not satisfy the physical constraints (6) and (7). Despite this, some repair work to the offspring is still required to maintain solution feasibility. Firstly, duplicated ships after crossover are removed from the offspring. These ships are removed from the original berths, while the newly acquired berth remains intact. This is followed by identifying missing ships in each of the offspring and reinserting them back into the chromosome. The newly acquired berth is excluded from this reinsertion process unless it is the only berth that can accommodate the ship considering the physical constraints (6) and (7). The probability that a ship is inserted into a particular berth is inversely proportional to the handling time of the ship at the berth. The ship is then inserted at a random position in the selected berth provided the insertion does not violate the physical constraints (6) and (7), otherwise another berth will be selected. This insertion process, where a ship has a higher chance of being inserted into a berth where it has a lower handling time, will be referred to as optimal berth insertion.

One of the advantages of berth-exchange crossover is that feasibility, in terms of the physical constraints (6) and (7), is easily maintained. Duplicated ships in each offspring are easily tracked by going through the ships in the newly acquired berth since only these ships can cause duplications. Each offspring inherits from both parents the relative order of ships within each berth since the removal of duplicated ships does not alter the order of ships. The reinsertion of missing ships provides some genetic variation.

3.4 Mutation

Mutation operators complement crossover operators in allowing a larger search space to be explored. In the MOEA, chromosomes are chosen to undergo mutation with a probability equal to the mutation rate. Mutation involves removing a number of ships, randomly selected based on the reinsertion rate, from the chromosome. These ships are then reinserted back into the chromosome based on the optimal berth insertion procedure described in the previous section.

3.5 Local search exploitation

Local exploitation can contribute to the intensification of the optimization results and is usually regarded as a complement to the evolutionary operators that mainly focus on global exploration.

The MOEA utilizes a local search operator aimed at reducing the number of crossings in solutions. The operator simply involves sorting the ships assigned to a berth in accordance to their priority orders. The operator is applied to all the berths in a chromosome and the order in which the berths are being operated by the heuristic is random. The solution is stored each time a berth is sorted. At the end of the entire operation on a particular chromosome, the number of solutions stored is equal to the number of berths in the chromosome. The pool of solutions is then decoded, evaluated and ranked based on the Pareto ranking scheme. The non-dominated solutions of the pool are inserted into the original population of chromosomes before the local search operator advances to operate on the next chromosome. After the entire original population has undergone local search, Pareto ranking is applied to the new population and the poorly ranked solutions will be removed from the population until the size of the population remains the same as before local search.

The local search operator designed is non-iterative in nature and does not compound to the computational intensity of the MOEA, which is a population-based search procedure and whose fitness evaluations are expensive due to the need to compute the values of the three considered objectives for each chromosome evaluated.

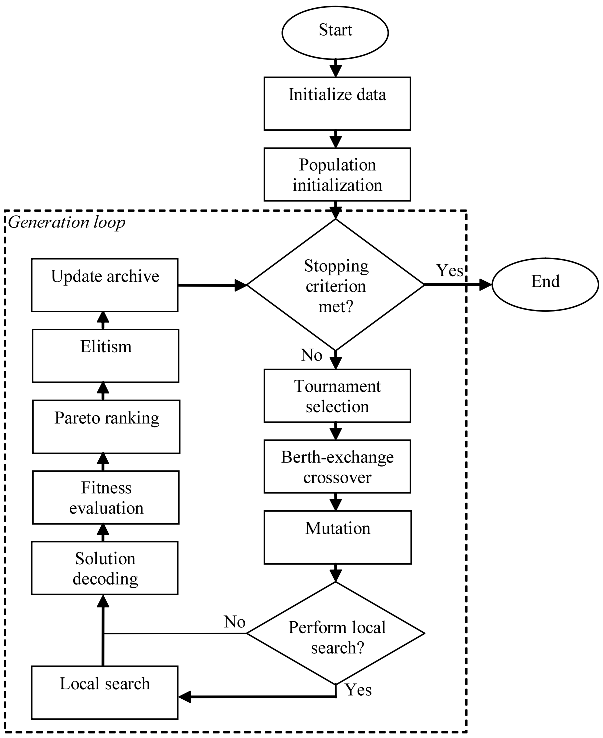
3.6 Algorithmic flow of MOEA

The algorithmic flow of the MOEA is shown in Fig. 8. At the start of the program, the berth data (berth length, water depth), ship data (ship length, ship draft, ship arrival time), and handling time data are loaded.

1. Initialization In the population initialization process, each chromosome is formed with the predefined number of berths. Ships are then inserted into the berths using the optimal berth insertion procedure that is also utilized in crossover and mutation.

2. Evaluation After the initial evolving population is formed, all the chromosomes are decoded using the solution decoding scheme. The fitness of the chromosomes are evaluated at the same time and after which, the chromosomes are ranked using the Pareto ranking scheme. Following the ranking process, an archive population is updated. The archive population has the same size as the evolving population and is used to store all the best solutions found during the search. The archive population updating process consists of a few steps. The evolving population is first appended to the archive population. All repeated chromosomes, in terms of the objective domain, are deleted. Pareto ranking is then performed on the remaining chromosomes in the population. The higher ranked (weaker) chromosomes are then deleted such that the size of the archive population remains the same as before the updating process. The evolving population remains intact throughout the updating process.

Fig. 8 Flowchart of MOEA



3. Genetic operations The binary tournament selection scheme is then performed. All the chromosomes in the evolving population are randomly grouped into pairs and from each pair, the chromosome with the lower rank is selected for reproduction. This procedure is performed twice to preserve the original population size. The genetic operators consist of berth-exchange crossover and mutation. To further improve the quality of berth schedules, local search is applied to the evolving and archive populations at regular intervals for better local exploitation in the evolutionary search.

4. Elitism A simple elitism mechanism is employed in the MOEA for faster convergence. The elitism strategy involves randomly picking a number of non-dominated solutions (5% of the population size) from the archive population. The chosen solutions then replace the worst ranked solutions in the evolving population.

This marks one complete generation of the MOEA and the evolution process iterates for a predefined number of generations.

4 Simulation results and analysis

The MOEA was programmed in C++ and simulations were performed on an Intel Pentium 4 3.2 GHz computer. Table 1 shows the parameter settings chosen after some preliminary experiments.

Since there is no commonly used benchmark for the BAP in the literature, many researchers have generated their own test problems, with a few of them using information

Table 1 Parameter settings for simulation study

Parameter	Value
Population size	200
Generation number	400
Crossover rate	0.8
Mutation rate	0.3
Reinsertion rate	0.1

Table 2 Test problem parameter settings

Parameter	Characteristic
Berth length	Uniformly between 350 to 700
Berth depth	Uniformly between 40 to 60
Ship length	Uniformly between 100 and 350
Ship draft	Uniformly between 30 and 60
Ship arrival	Exponential interval with mean 12
Ship handling time	2-Erlangian distribution

Table 3 Characteristics of test problems

Test problem	Number of berths	Number of ships	Priority order
BAP5 × 100F	5	100	First-come-first-serve
BAP5 × 100L	5	100	Last-come-first serve
BAP5 × 200F	5	200	First-come-first-serve
BAP5 × 200L	5	200	Last-come-first serve
BAP10 × 100F	10	100	First-come-first-serve
BAP10 × 100L	10	100	Last-come-first serve
BAP10 × 200F	10	200	First-come-first-serve
BAP10 × 200L	10	200	Last-come-first serve

from their ports of study. The test problems in this paper are generated randomly but systematically. Ship arrivals are generated using an exponential distribution while ship handling times are based on a 2-Erlangian distribution. Imai et al. (2001) obtained these distributions from their survey on the port of Kobe. Based on the parameter settings in Table 2, eight test problems are generated. The characteristics of these problems are given in Table 3. Two extreme priority policies are experimented. First-come-first-serve (FCFS) is where ships are given priority in order of increasing arrival time, while last-come-first-serve (LCFS) is where the last ship that arrives at the port is given top priority. Although the LCFS priority order represents an impossible hypothetical port management policy, it provides a contrasting situation to the FCFS policy and is able to reveal certain characteristics of the MOEA. BAP5 × 100F and BAP5 × 100L are used for developing the algorithm, while the rest of the test problems are used to validate the performance of the proposed MOEA.

The subsequent sections present extensive simulation results and analysis of the proposed MOEA. Sections 4.1, 4.2, and 4.3, respectively, study the effects that the three primary features of local search exploitation, solution decoding scheme, and optimal berth insertion have on the quality of the generated berth schedules. The optimization performance of the developed MOEA is then validated against a simple MOEA in Sect. 4.4.

4.1 Effects of local exploitation on quality of berth schedules

The MOEA incorporates local search exploitation to complement the evolutionary operators of berth-exchange crossover and mutation, which focus on global evolutionary optimization. As described in Sect. 3.5, although the local search operator targets at reducing the number of crossings in solutions, the addition of the Pareto ranking scheme in the operator ensures that it accounts for the multi-objective nature of the problem. This section studies how the frequency of local search can affect the performance of the MOEA. At the same time, it demonstrates the effectiveness of local search in reducing the number of crossings in solutions, as well as its other implications on the quality of solutions.

Simulations were conducted by varying the frequency at which local search is performed. LS25, LS50, LS100, and LS200 are the MOEA settings where local search is applied to the evolving and archive populations every 25, 50, 100, and 200 generations, respectively. NLS is the setting which does not make use of local search at all. Ten independent runs of each of the settings were conducted on BAP5 × 100F to obtain statistical results.

The convergence traces of the three objectives of makespan, waiting time, and number of crossings for the five local search settings are plotted in Fig. 9. The convergence traces show

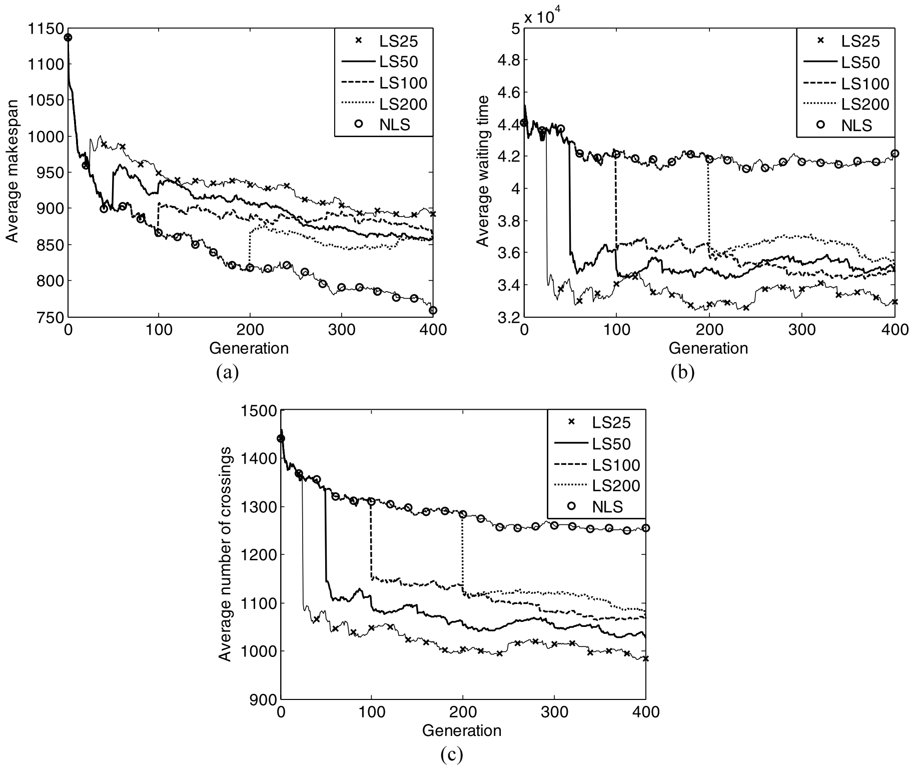


Fig. 9 (**a**) Average makespan, (**b**) average waiting time, and (**c**) average number of crossings of non-dominated solutions for different local search settings on BAP5 × 100F

the change in the respective objective values, averaged over all the non-dominated solutions in the archive population, over the generations. The values are further averaged over the 10 simulation runs performed. Figure 9(c) shows the effectiveness of local exploitation in the MOEA in reducing the number of crossings in solutions. The local search operator causes dips in the number of crossings whenever it is applied to solutions. Comparing the convergence traces for the different local search settings, the dips in the number of crossings get more prominent with the increase in frequency of application of the local search heuristic.

Another observation is that the dips in number of crossings coincide with the dips in waiting time in Fig. 9(b) and the rises in makespan in Fig. 9(a). As the local search operator tries to reduce the number of crossings in each solution, it indirectly reduces the waiting time but increases the makespan of the solution. This seems to suggest that the three objectives are related to one another. It appears that the objectives of makespan and waiting time are conflicting with each other, i.e. any attempt to minimize either of the objectives will cause the other objective to increase.

The same simulations were also conducted on BAP5 $\times$ 100L and the corresponding convergence traces are plotted in Fig. 10. Like in Fig. 9(c), in Fig. 10(c), the local search operator causes dips in the number of crossings whenever it is applied to solutions. While this observation is expected since the operator is specifically designed to reduce the number of crossings in solutions, Fig. 10 does provide an interesting result. In contrast to the observation in Fig. 9 that a reduction in the number of crossings in a solution causes an increase in

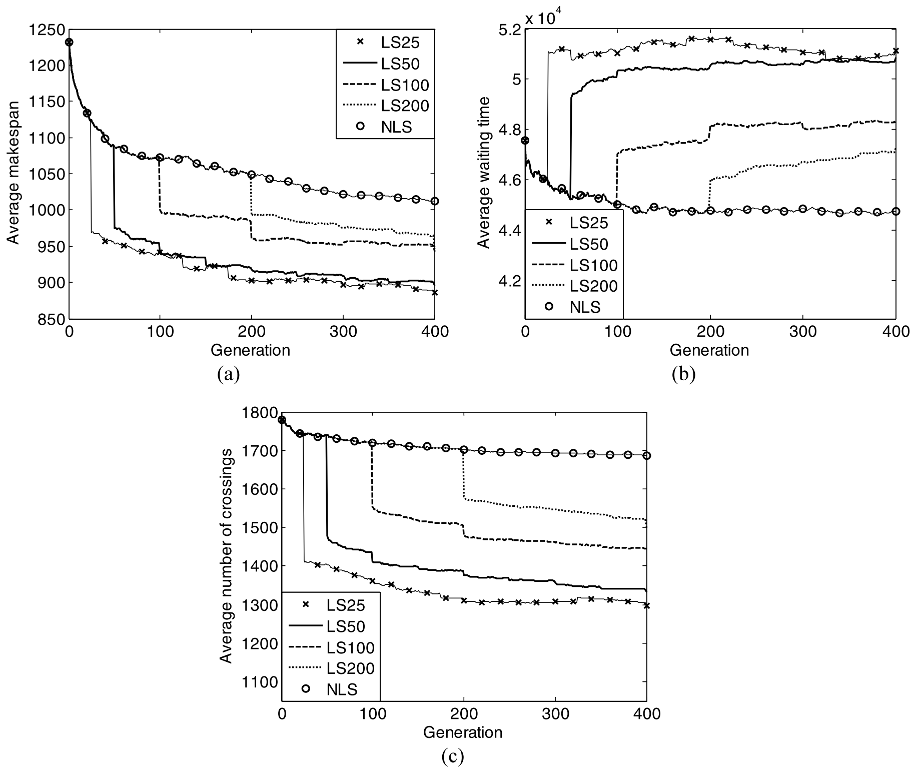


Fig. 10 (**a**) Average makespan, (**b**) average waiting time, and (**c**) average number of crossings of non-dominated solutions for different local search settings on BAP5 $\times$ 100L

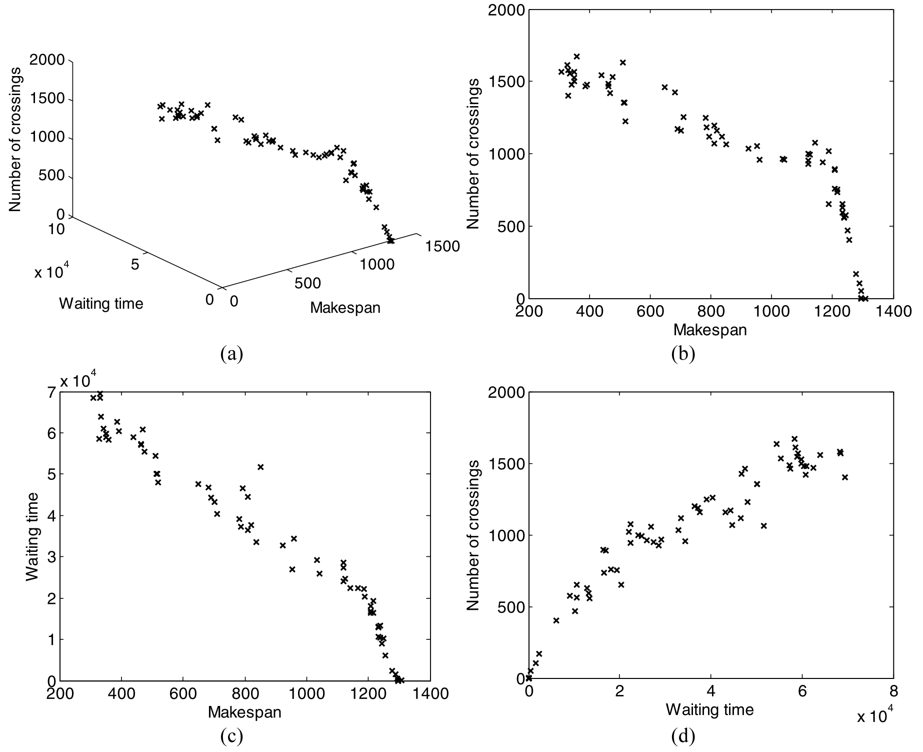


Fig. 11 Pareto front for a random run of LS50 on BAP5 × 100F

makespan and a decrease in waiting time, Fig. 10 shows the exact opposite result, i.e. the reduction in the number of crossings in a solution leads to a decrease in makespan and an increase in waiting time. While the results again show that the objectives of makespan and waiting time are conflicting with each other, their relationships with the third objective have changed.

To further confirm the relationships between the three objectives in the BAP, the Pareto fronts, each of which being made up of all the non-dominated solutions in the archive population, for a random run of LS50 on BAP5 × 100F and BAP5 × 100L are plotted in Figs. 11(a) and 12(a), respectively. Separate two-dimensional graphs are also plotted for clarity in analyzing the relationships between the objectives. The plots in Figs. 11(c) and 12(c) confirm that regardless of the priority policy adopted by the port, the objectives of makespan and waiting time are conflicting with each other. On hindsight, this relation between the two objectives can be explained. In minimizing makespan, the port should delay berthing ships even when they have arrived at the port to reduce the berth idle time in between berthing of ships. In this way, ships would be waiting in the port and can berth as soon as their preceding ships have been serviced. This practice will, of course, incur the dissatisfaction of ship operators since their ships have to spend a longer time waiting at the port. The plots in Figs. 11(b), 11(d), 12(b), and 12(d) show that there is generally no fixed relation between the number of crossings in a solution and the other two objectives. However, Fig. 11(d) shows that the FCFS service policy leads to a proportional relationship between number of crossings and waiting time, i.e. a decrease in the number of crossings in

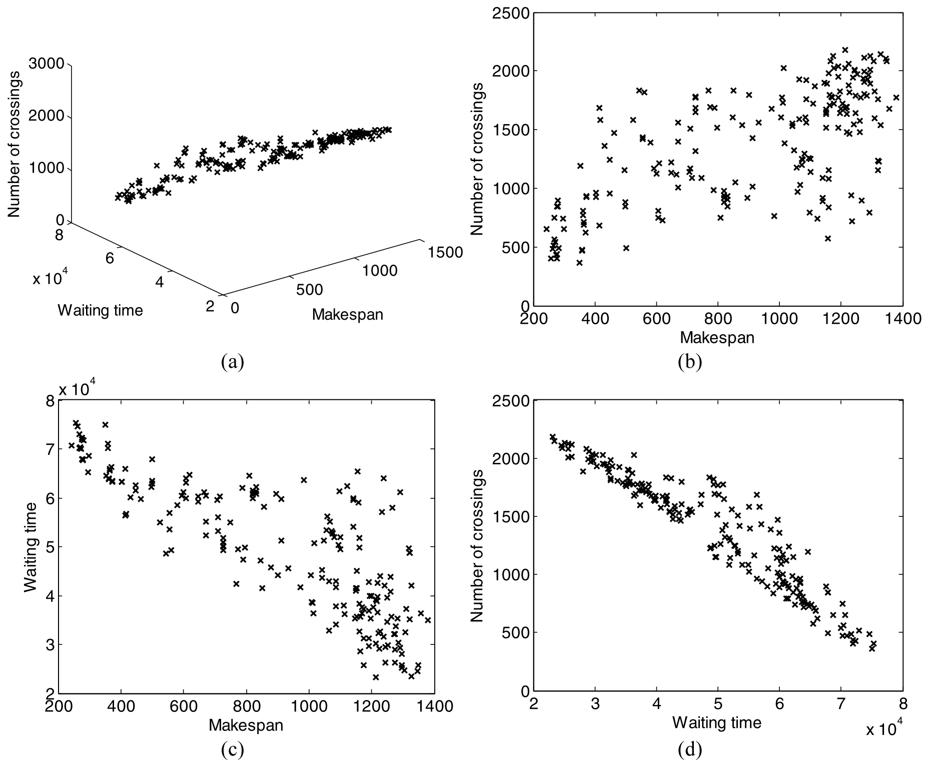


Fig. 12 Pareto front for a random run of LS50 on BAP5 $\times$ 100L

a solution leads to a decrease in waiting time, while Fig. 12(d) demonstrates that the LCFS policy leads to a conflicting relationship between the two objectives. Given the conflicting relation between makespan and waiting time, the relation between makespan and number of crossings for the LCFS policy is proportional, which can be vaguely observed in Fig. 12(b) since most of the solutions are situated at the bottom left and top right of the plot. This relation between makespan and number of crossings suggests that a port targeting to reduce its makespan should adopt a LCFS service policy, while the FCFS service policy benefits ship operators more in terms of lower waiting times for their ships.

The previous results have established the relationships between the three objectives of makespan, waiting time, and number of crossings. Judging from the intricate relationships between the three objectives, it can be concluded that the BAP is inherently a multi-objective problem which needs to be solved from the perspectives of both port and ship operators. In this aspect, the MOEA, which is able to generate a Pareto set of berth schedules from which a solution that can satisfy both port and ship operators to acceptable degrees can be selected for implementation, can perform satisfactorily.

Unlike single-objective optimization which produces a single optimal solution such that solution quality can be easily compared based on the considered objective, the solution to multi-objective optimization exists in the form of the Pareto optimal set. As such, in comparing the performance of the different local search settings in this section, it is required to compare the optimality of their respective Pareto fronts. The optimality of Pareto fronts are usually compared based on the proximity and diversity with respect to the optimal Pareto front (Deb 2001; Bosman and Thierens 2003). Proximity indicates how close a Pareto front is from the optimal Pareto front, while diversity indicates how well-distributed and diverse the space along the Pareto front is covered with solutions. There are many multi-objective performance indicators in the literature measuring the proximity and diversity of a Pareto front. While some performance indicators require the knowledge of the optimal Pareto front, some do not. The former are a better indication of multi-objective performance since the optimal Pareto front provides a basis for comparison. However, it is often the case that the optimal Pareto front is unknown and it is simply intractable to compute it, especially in large real-world combinatorial problems such as the BAP. As such, four performance indicators that do not require the knowledge of the optimal Pareto front have been chosen in this paper to compare multi-objective optimization performance.

The first performance indicator is the coverage function (C) (Zitzler and Thiele 1999) which measures the proximity of a Pareto front. It is a binary quality measure which compares the dominance relationship between pairs of solution sets or Pareto fronts (Zitzler et al. 2003). Given a pair of solution sets (A, B), the coverage function $C(A, B)$ returns the fraction of solutions in B that are weakly dominated by at least one solution in A. As opposed to the definition of dominance given in Sect. 2.1, which is used for ranking solutions in the MOEA, weak dominance implies that a solution dominates another solution even if they are equal in terms of all the considered objectives. As such, if $C(A, B)$ returns a value of 1, it means that all the solutions in B are dominated by or equal to the solutions in A.

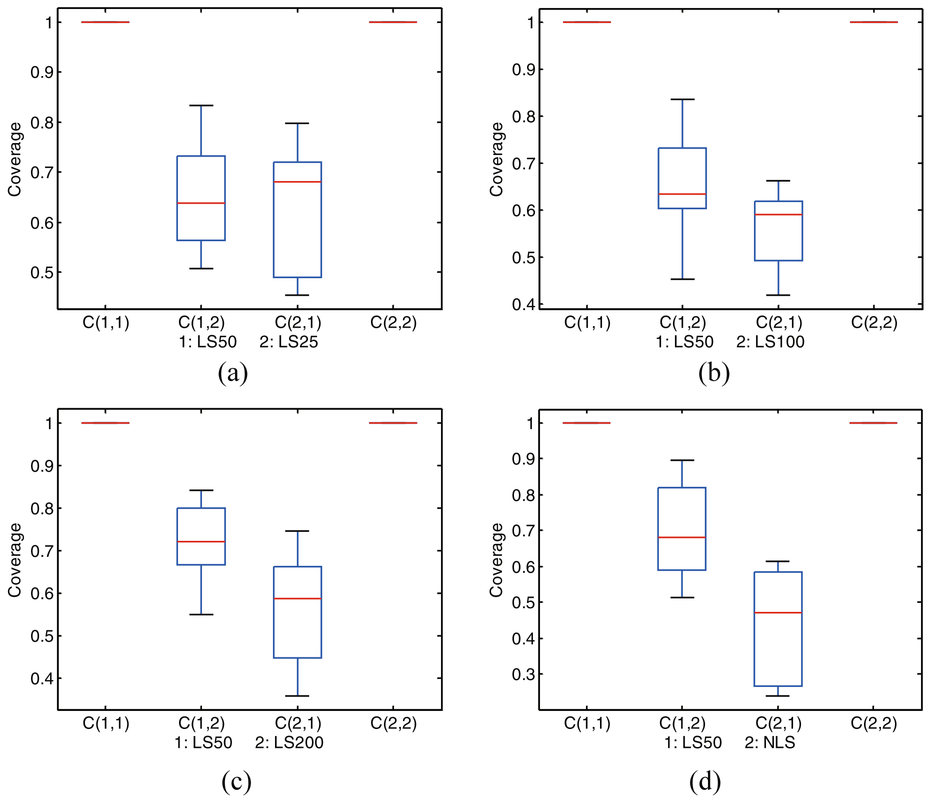


Fig. 13 Coverage results for different local search settings on BAP5 × 100F

The other extreme case, where $C(A, B)$ returns a value of 0, implies that none of the solutions in B is weakly dominated by any of the solutions in A. It should be highlighted that both $C(A, B)$ and $C(B, A)$ have to be considered for a complete performance assessment. If $C(A, B)$ returns a high value and $C(B, A)$ returns a low value, it can be implied that the Pareto front made up of the solutions in A is closer to the optimal Pareto front than that made up of the solutions in B.

In comparing the performance of the local search settings, due to the binary nature of the coverage function, LS50 is chosen as the basis for comparison. Since 10 independent runs of each of the settings were conducted, the comparisons are based on corresponding runs of each pair of settings, i.e. the Pareto front obtained by run number 1 of a setting is compared only with the Pareto front obtained by run number 1 of the other setting, as they share the same random number seed. The results of these comparisons are then represented in box plots and are shown in Figs. 13 and 14. Each box plot represents the distribution of the values returned by the coverage function for the 10 comparisons made for each pair of settings where the horizontal line within the box encodes the median, and the upper and lower ends of the box are the upper and lower quartiles, respectively. The two horizontal lines beyond the box give an indication of the spread of the data. A plus sign outside the box represents an outlier.

Comparing the coverage results in Fig. 13, on BAP5 × 100F, the MOEA performs better with the increase in frequency of application of the local search heuristic. In Figs. 13(b),

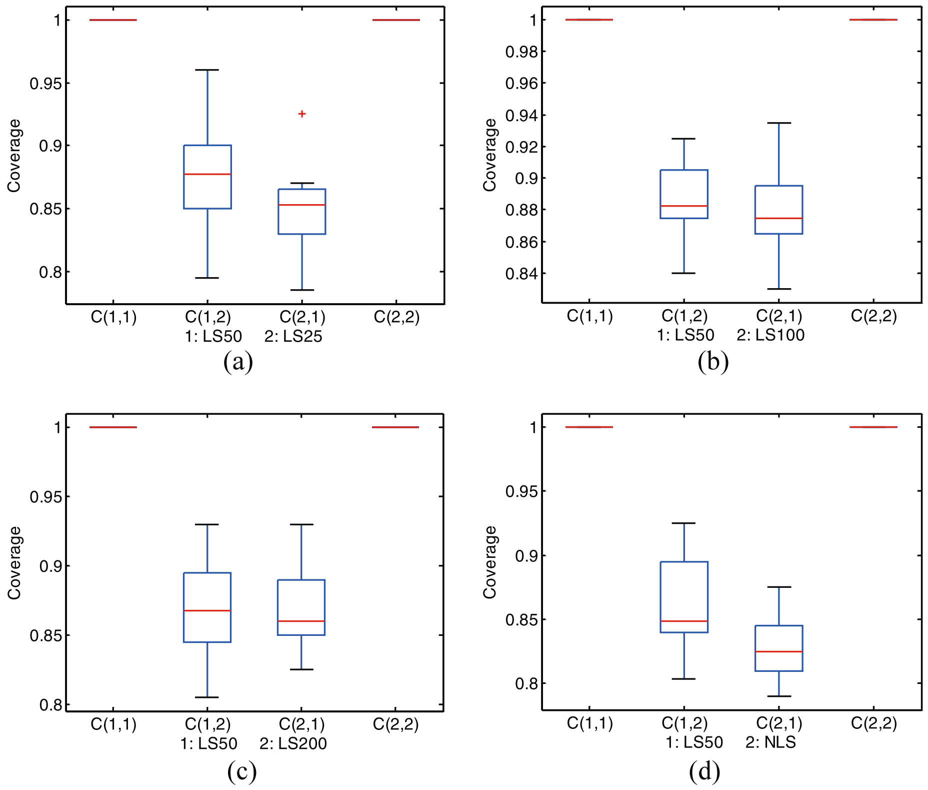


Fig. 14 Coverage results for different local search settings on BAP5 × 100L

13(c), and 13(d), the difference between the medians of the coverage results gets smaller as the frequency of local search is increased with LS25 slightly surpassing the performance of LS50 in Fig. 13(a). In Fig. 14, LS50 generally performs better than the other local search settings on BAP5 $\times$ 100L.

Three performance indicators are used to measure the diversity of a Pareto front. The first is an adaptation of the popular maximum spread measure (Zitzler et al. 2000) which indicates the maximum range of the optimal Pareto front that is being covered by the generated solutions. Since the measure assumes the knowledge of the optimal Pareto front, an alternative measure, which computes the volume in the objective domain covered by the generated solutions, is used. The measure, referred to as spread, is defined in (15). A larger spread value implies that the solutions in the Pareto front cover a wider range of values of each of the objectives, indicating a more diverse solution set.

$$\begin{aligned}\text{Spread} = {} & (\text{Makespan}_{\text{max}} - \text{Makespan}_{\text{min}}) \times (\text{Waiting time}_{\text{max}} - \text{Waiting time}_{\text{min}}) \\ & \times (\text{Number of crossings}_{\text{max}} - \text{Number of crossings}_{\text{min}}) \qquad (15)\end{aligned}$$

The next performance indicator is spacing which measures the variance of the distance of each of the solutions in the Pareto front from its nearest neighbor. Distance is measured with respect to the Euclidean distance in the three-dimensional objective space. A low spacing

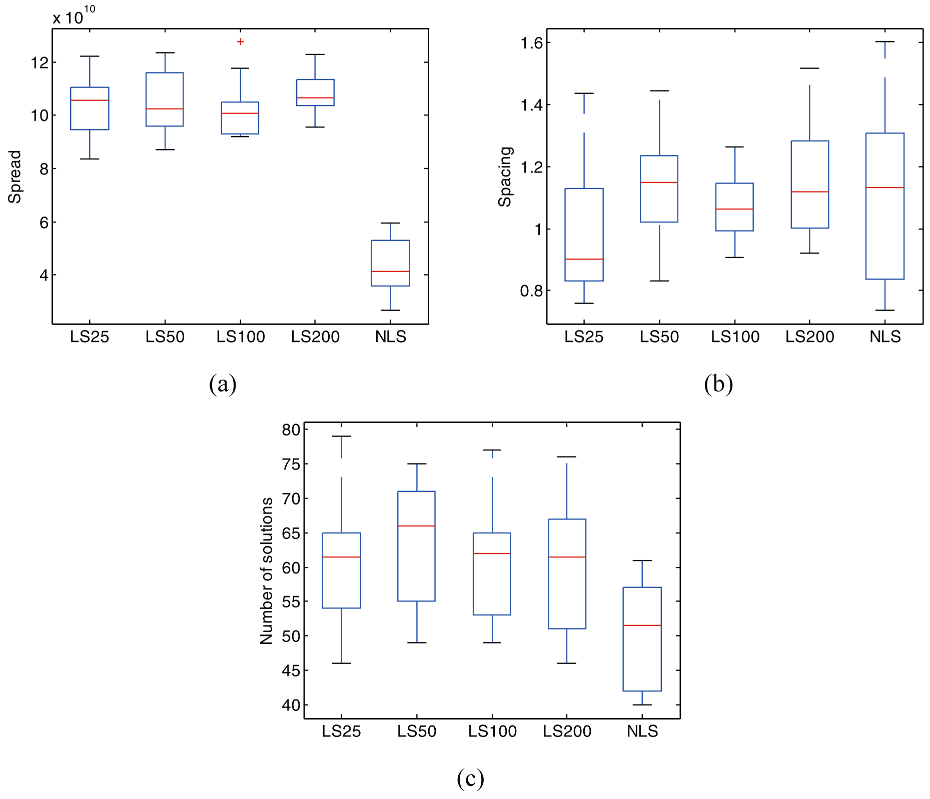


Fig. 15 (**a**) Spread, (**b**) spacing, and (**c**) number of non-dominated solutions for different local search settings on BAP5 $\times$ 100F

value implies that the solutions are more evenly distributed over the entire Pareto front. The last indicator for measuring the diversity of a Pareto front is simply the number of solutions that form the Pareto front. It gives an idea of how effective the algorithm is in generating desired solutions. In order to conclude that a Pareto front is diverse, it has to score well in all the three performance indicators. If an algorithm performs well in terms of spread and number of solutions in the Pareto front but does badly in spacing, it only means that there are many huge gaps in the Pareto front which the algorithm has failed to explore.

The performance of the five local search settings for the three diversity performance indicators are again represented in box plots in Figs. 15 and 16, which show the distributions of the respective indicator values over the 10 independent simulation runs.

From the results in Figs. 15 and 16, it is clear that local search is beneficial to the MOEA. NLS gives poor diversity performance as the generated Pareto front has a significantly lower spread, larger spacing, and lower number of solutions compared to those generated by the other settings which make use of local search. LS50 generally performed well for the three diversity performance indicators. Coupled with its favorable proximity performance as seen in Figs. 13 and 14, LS50 is selected as the default local search setting for any further analysis of the MOEA.

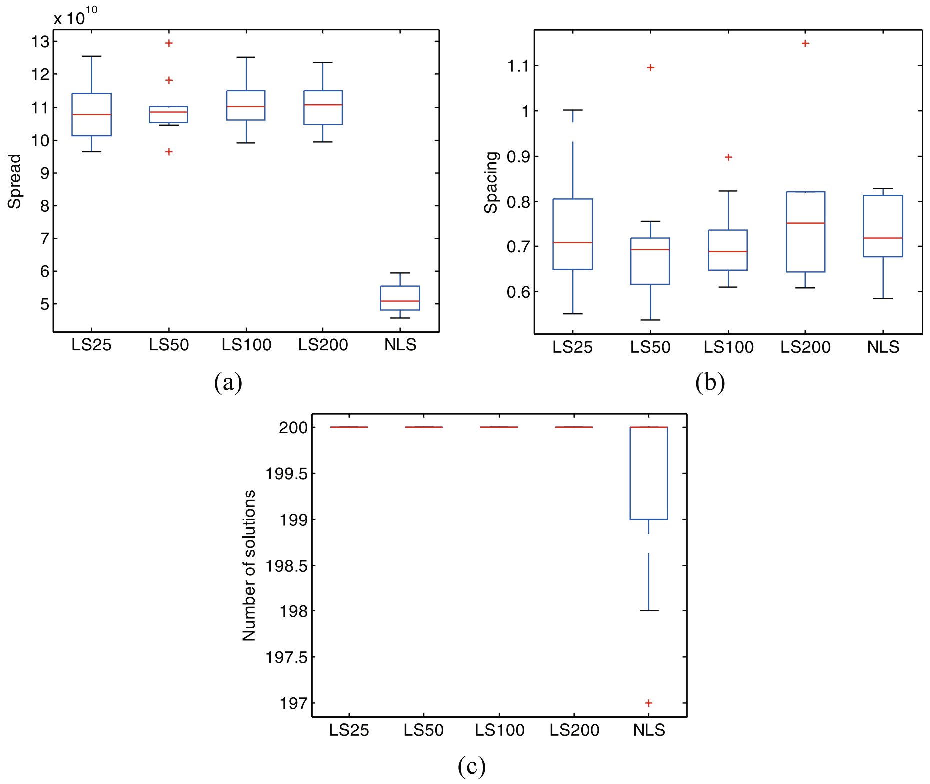


Fig. 16 (**a**) Spread, (**b**) spacing, and (**c**) number of non-dominated solutions for different local search settings on BAP5 $\times$ 100L

4.2 Effects of solution decoding schemes on quality of berth schedules

Two solution decoding schemes have been introduced in Sect. 3.2 to decode chromosomes into berth schedules for fitness evaluation. It has been highlighted that a chromosome may have two different sets of objective values depending on the decoding scheme applied. This section proposes a hybrid solution decoding scheme which makes use of both decoding schemes, as well as studies the effects that the two decoding schemes have on the quality of berth schedules.

Simulations were conducted using five different MOEA settings. BOD is the setting which uses solely the berthing order decoding scheme for decoding solutions, while AOD is the setting that uses only the assignment order decoding scheme. A hybrid solution decoding scheme, where each solution has a certain chance to be decoded by either of the decoding schemes, is also tested. Hybrid25, Hybrid50, and Hybrid75, respectively, are the settings where each solution has a 25%, 50%, and 75% chance of being operated by the assignment order decoding scheme, otherwise it will be operated by the berthing order decoding scheme. Like in the previous section, 10 simulation runs of each of the five settings were performed on BAP5 $\times$ 100F and BAP5 $\times$ 100L.

The convergence traces of the three objectives, averaged over the non-dominated solutions and over the 10 simulation runs, for the five settings on BAP5 $\times$ 100F and BAP5 $\times$ 100L are plotted in Figs. 17 and 18, respectively. It can be seen in Fig. 17 that AOD and BOD

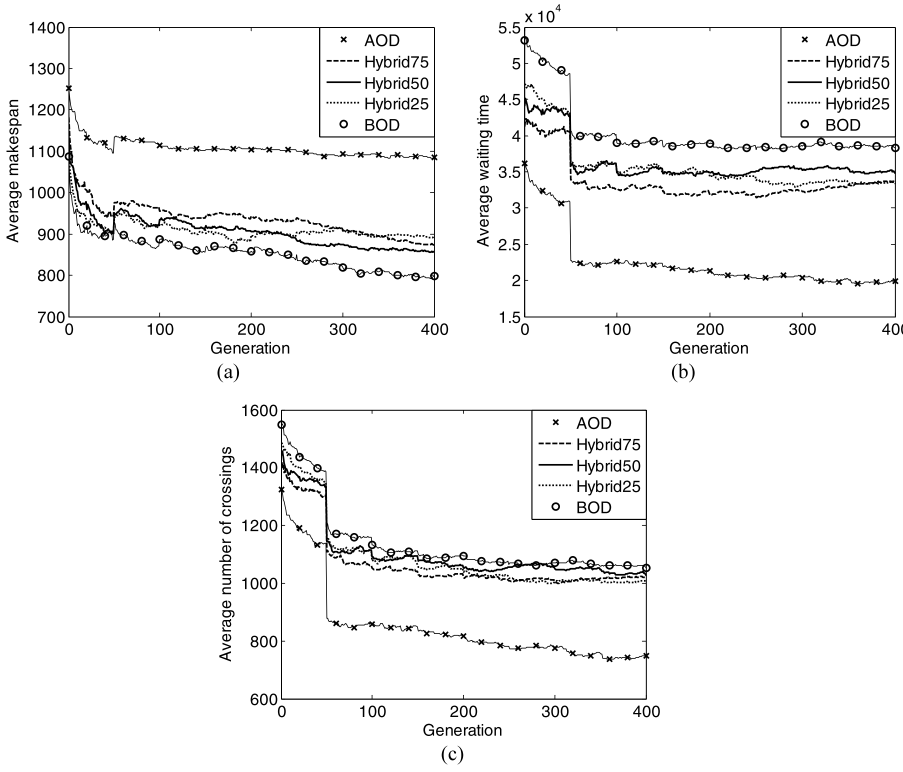


Fig. 17 (**a**) Average makespan, (**b**) average waiting time, and (**c**) average number of crossings of non-dominated solutions for different solution decoding settings on BAP5 $\times$ 100F

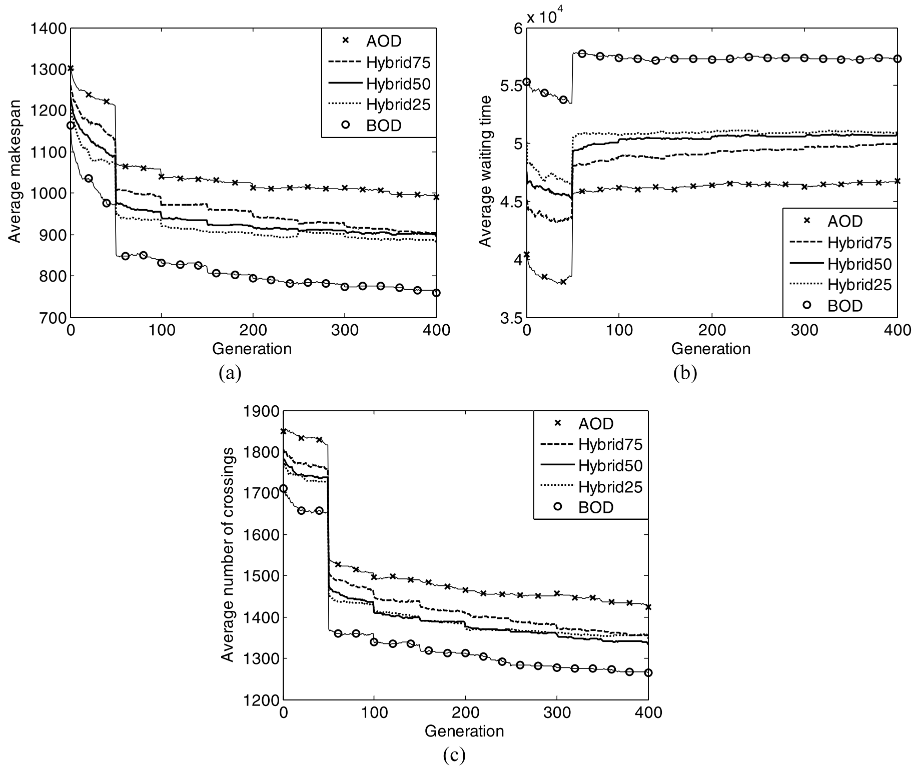


Fig. 18 (**a**) Average makespan, (**b**) average waiting time, and (**c**) average number of crossings of non-dominated solutions for different solution decoding settings on BAP5 × 100L

provide two extreme results. While AOD has a tendency of generating solutions with high makespans and low waiting times, BOD tends to concentrate on solutions with low makespans and high waiting times. The same results are also observed in Fig. 18 for the simulations on BAP5 × 100L. In terms of number of crossings, AOD achieves better results than BOD in Fig. 17(c) but performs worse in Fig. 18(c). Given the mixed effects that the two settings have on the number of crossings in solutions, it can be inferred that the type of solution decoding scheme does not have any direct effect on the objective. Rather, the type of decoding scheme has a direct impact on the other two objectives of makespan and waiting time with the assignment order decoding scheme churning berth schedules with high makespans and low waiting times and the berthing order decoding scheme decoding solutions into schedules with low makespans and high waiting times. Any observable effect on the number of crossings in solutions is due to the underlying relationships between the three objectives for FCFS and LCFS problems that have been identified in the previous section. The effect would not be obvious if a different priority policy were adopted. One probable explanation for the berthing order decoding scheme generating berth schedules with lower makespans and higher waiting times can be made with reference to Fig. 5. In Fig. 5(a), the berth schedule generated by the berthing order decoding scheme has a lower makespan compared to that generated by the assignment order decoding scheme in Fig. 5(b). The berthing order decoding scheme states that succeeding ships cannot be berthed earlier than ship 4 even though they arrive at the port earlier than ship 4. This results in the berthing times of

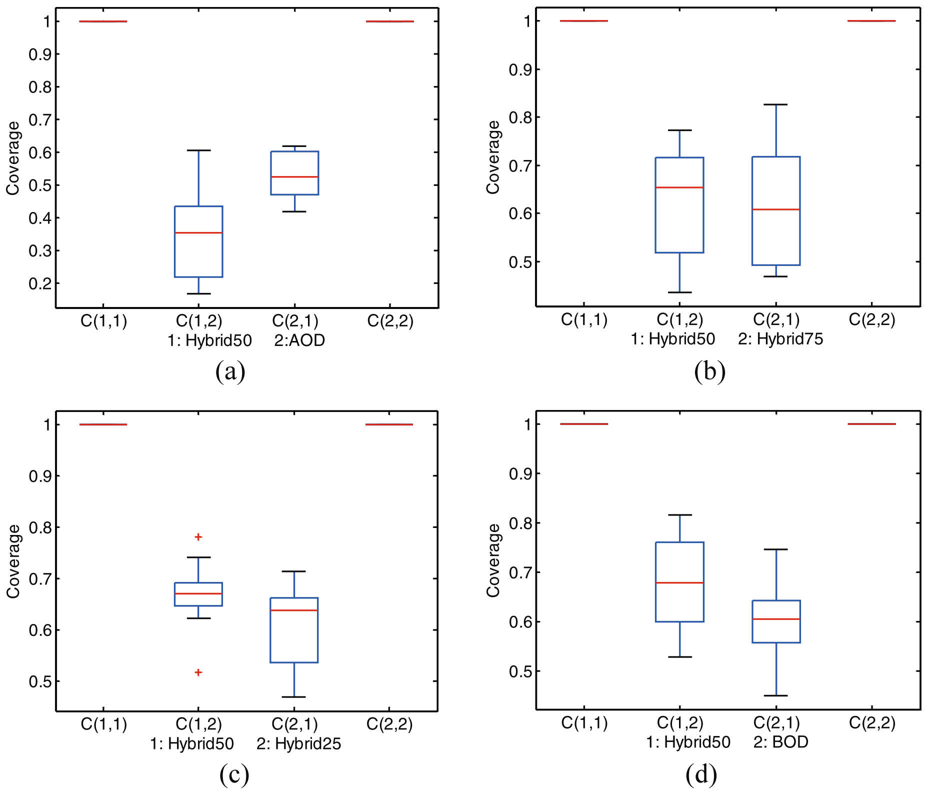


Fig. 19 Coverage results for different decoding scheme settings on BAP5 × 100F

ships 1, 2, 3, and 5 to be pushed back, leading to a lower makespan. However, these ships would incur longer waiting times compared to their counterparts in Fig. 5(b).

Having seen the contrasting effects that the two solution decoding schemes have on the quality of berth schedules, it makes sense to use a hybrid decoding scheme that makes use of both decoding schemes. From Figs. 17 and 18, it can be seen that the three settings which make use of the hybrid decoding scheme provide intermediate results within the limits set by the extreme settings of AOD and BOD.

In order to compare the performance of the five settings, the four performance indicators introduced in the previous section are used. Hybrid50 is used as the basis of comparison for computing the coverage results. The performance comparison results are shown in Figs. 19–22.

In general, the hybrid settings show better proximity and diversity results compared to AOD and BOD. There exists an abnormality though in Fig. 19(a), where AOD obtained a better coverage result over Hybrid50. In order to investigate the abnormality, the Pareto fronts for a random run of the two settings on BAP5 × 100F are plotted in Fig. 23.

A comparison of the Pareto fronts of Hybrid50 and AOD in Fig. 23 reveals a glaring deficiency in AOD. The setting is unable to locate any solution with a makespan lower than 600. It is obvious that the set of solutions generated by AOD is not as complete as that generated by Hybrid50. This explains the setting's poor performance in terms of the diversity performance indicators of spread and number of generated non-dominated solutions in Figs. 21(a)

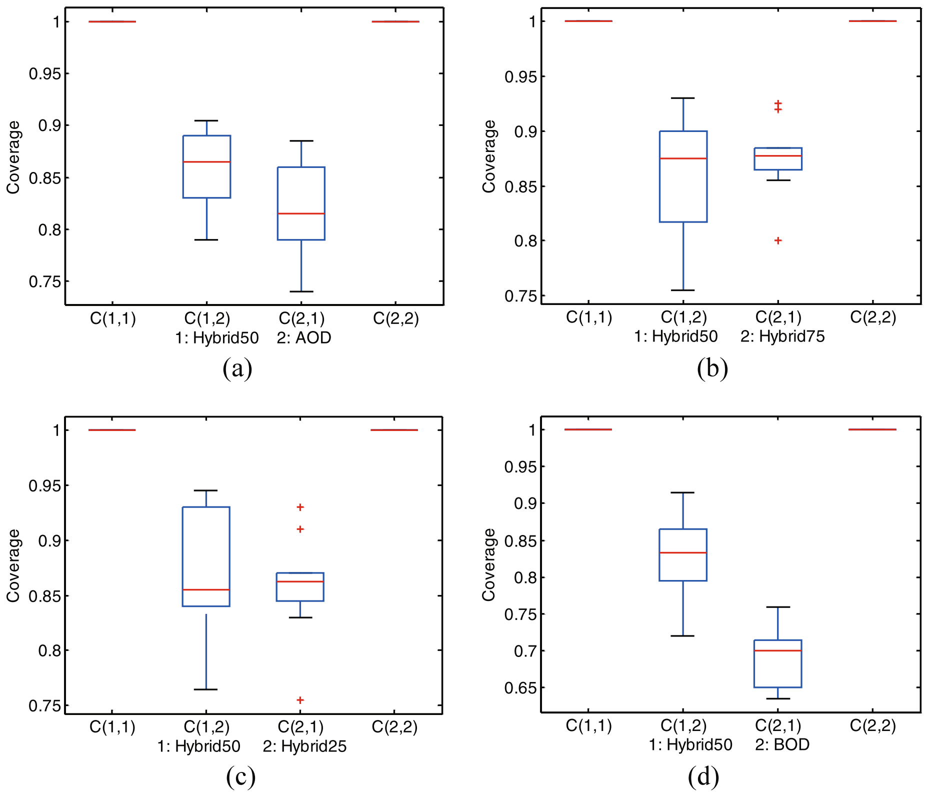


Fig. 20 Coverage results for different decoding scheme settings on BAP5 × 100L

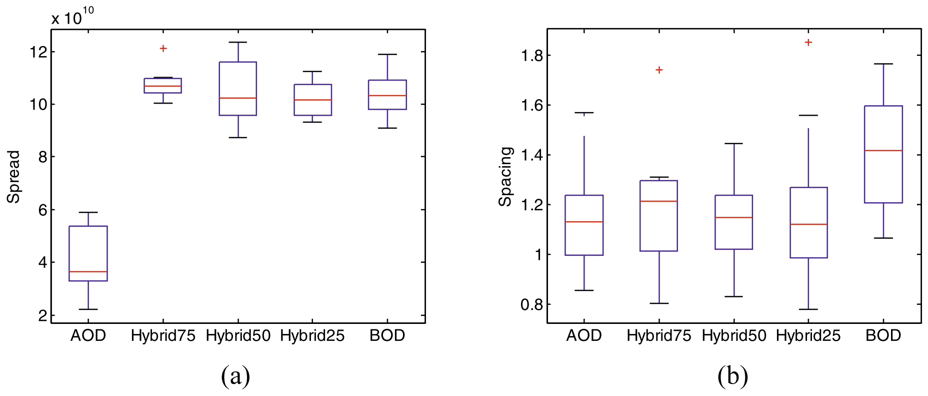
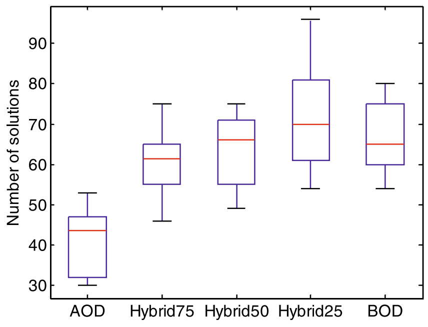


Fig. 21 (**a**) Spread, (**b**) spacing, and (**c**) number of non-dominated solutions for different decoding scheme settings on BAP5 × 100F

and 21(c), respectively. A likely reason explaining AOD's superior proximity performance over Hybrid50 can be observed in Fig. 23(c). Most of the solutions generated by AOD are able to dominate and at the same time, not being dominated by solutions generated by Hy-

Fig. 21 (*Continued*)



(c)

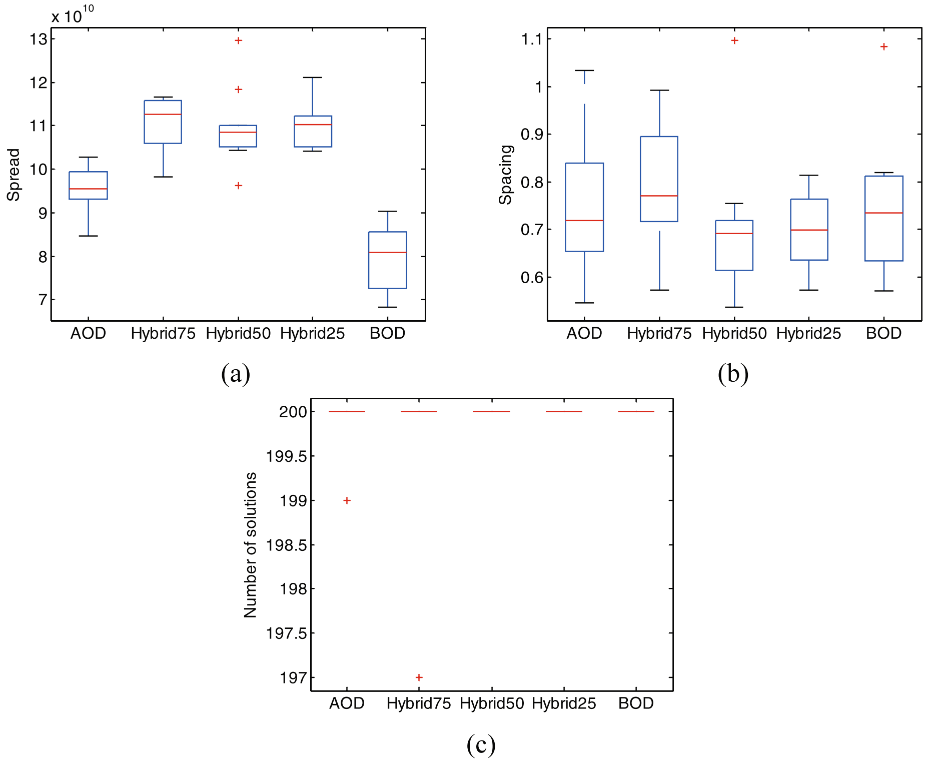


Fig. 22 (**a**) Spread, (**b**) spacing, and (**c**) number of non-dominated solutions for different decoding scheme settings on BAP5 $\times$ 100L

brid50 in terms of the two objectives of makespan and waiting time. It is likely that AOD has been spending its search efforts on other areas of the search space instead of locating low makespan solutions. In order to confirm this hypothesis, the search spaces in the objective domain explored by Hybrid50, AOD, and BOD are plotted in Fig. 24. Each point in the plots is a point in the objective domain that has been found by the respective settings during the simulation run. For clarity in analyzing the sizes of the search spaces explored by the three

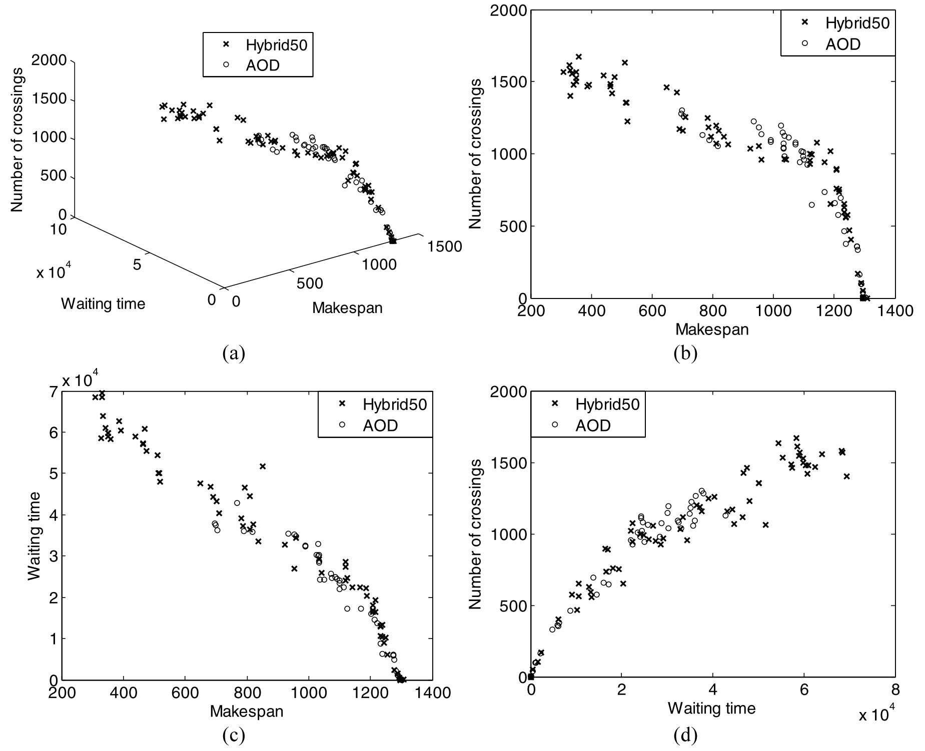


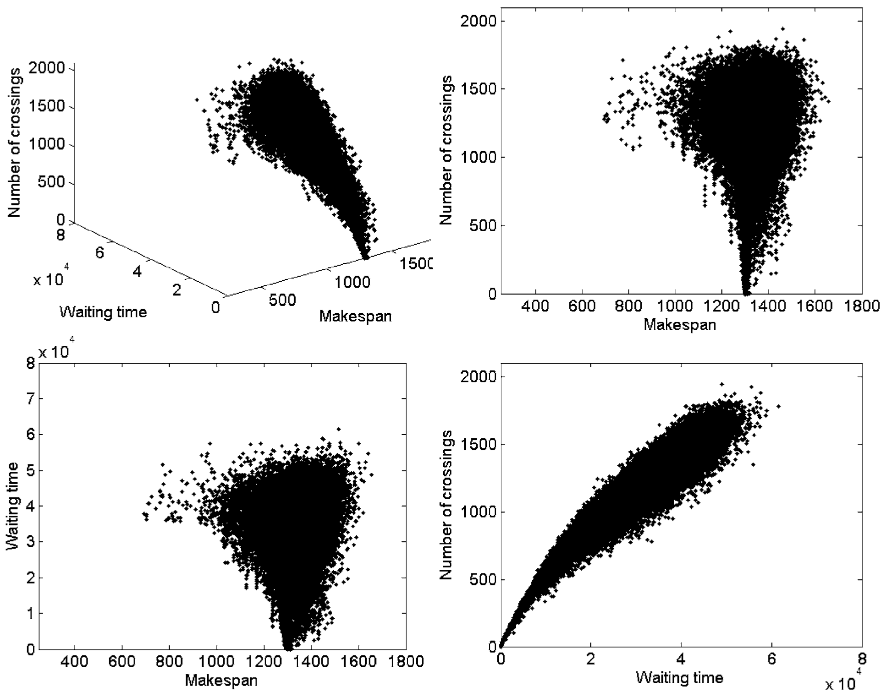
Fig. 23 Pareto fronts for a random run of Hybrid50 and AOD on BAP5 × 100F

settings, separate two-dimensional graphs are also plotted and the range of each of the axes is kept consistent throughout the plots.

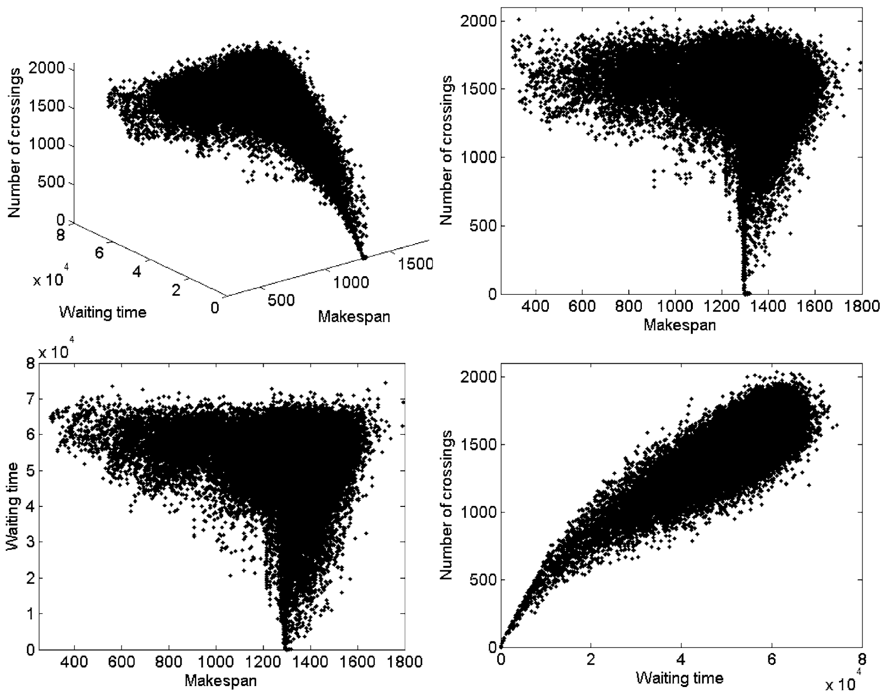
Comparing the search spaces of AOD and BOD in Figs. 24(a) and 24(b), it can be observed that certain parts of the search space that AOD has explored have been left out by BOD and vice versa. To allow a better visual comparison, the corresponding two-dimensional search space plots are superimposed onto each other in Fig. 25. Judging from the search spaces that both settings have left unexplored, it is quite clear that the search space explored by Hybrid50 in Fig. 24(c) is a union of the search spaces covered by AOD and BOD. The hybrid setting is able to benefit from the complementary behavior of the two solution decoding schemes, which allows a larger search space to be explored. This advantage translates into better proximity and diversity results as have been observed in Figs. 19–22. Since the proximity and diversity results for the three hybrid settings are relatively comparable, Hybrid50 is chosen as the default setting for any subsequent analysis of the MOEA.

4.3 Effects of optimal berth insertion on quality of berth schedules

Optimal berth insertion is utilized during population initialization, berth-exchange crossover, and mutation to insert ships into chromosomes. The insertion procedure gives each ship a higher chance of being inserted into a berth where it has a lower handling time. This section presents the performance of the MOEA with and without the insertion procedure. In the

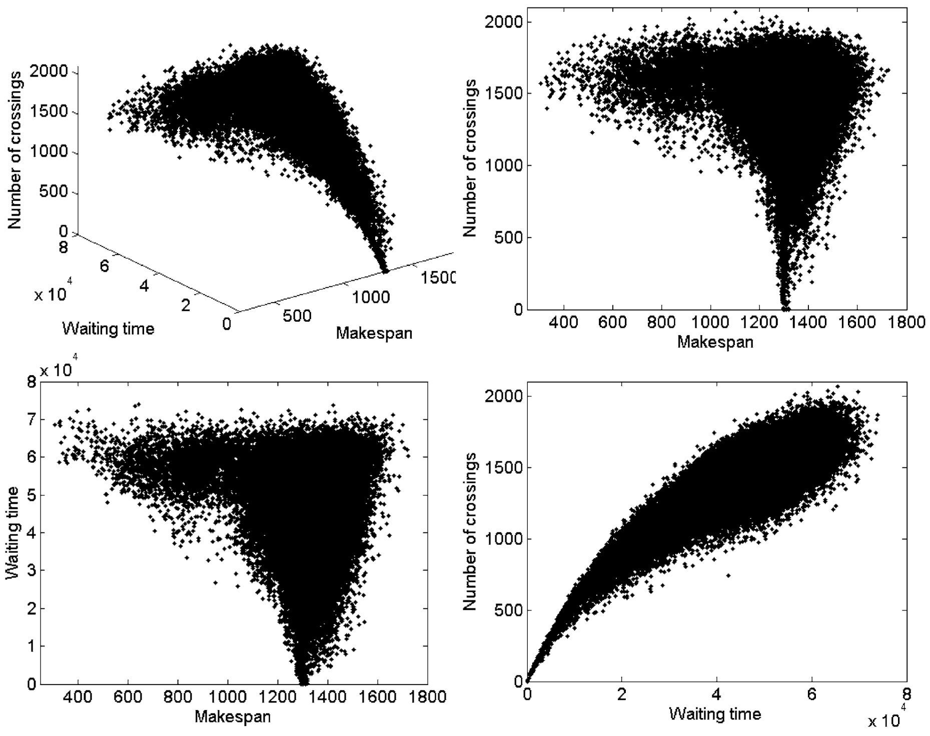


(a) AOD



(b) BOD

Fig. 24 Comparison of search spaces for different decoding scheme settings on BAP5 $\times$ 100F



(c) Hybrid50

Fig. 24 (*Continued*)

case where optimal berth insertion is not used, each ship has equal chance of being inserted into any of the berths. This setting will be known as RAND. Ten independent simulation runs of the MOEA and RAND were performed on BAP5 × 100F and BAP5 × 100L. Since the optimal berth insertion procedure targets to minimize the handling times of ships, the convergence traces of the total handling time incurred by the entire fleet of ships, averaged over the non-dominated solutions and over the 10 simulation runs, for the two settings on BAP5 × 100F and BAP5 × 100L are plotted in Fig. 26.

It is obvious from Fig. 26 that the optimal berth insertion procedure has achieved its aim of reducing the handling times of ships. In most situations, the reduction in handling time translates directly to a reduction in makespan and waiting time since handling time is a component of the two objectives. The lower average handling time for the MOEA at generation 0 shows the positive effect that optimal berth insertion has in population initialization. The steeper decline in average handling time for the MOEA in the initial stages of evolution is due to the incorporation of optimal berth insertion in berth-exchange crossover and mutation. To compare the multi-objective optimization performance of the two settings, the four proximity and diversity performance indicators are computed for the Pareto fronts generated by the two settings. The comparison results are plotted in Figs. 27 and 28.

From Figs. 27(a) and 28(a), it can be seen that the MOEA is superior to RAND in terms of the proximity performance measure. However, the three performance metrics of spread, spacing, and number of generated non-dominated solutions indicate that the two settings have comparable diversity performance. This finding implies that the optimal berth insertion

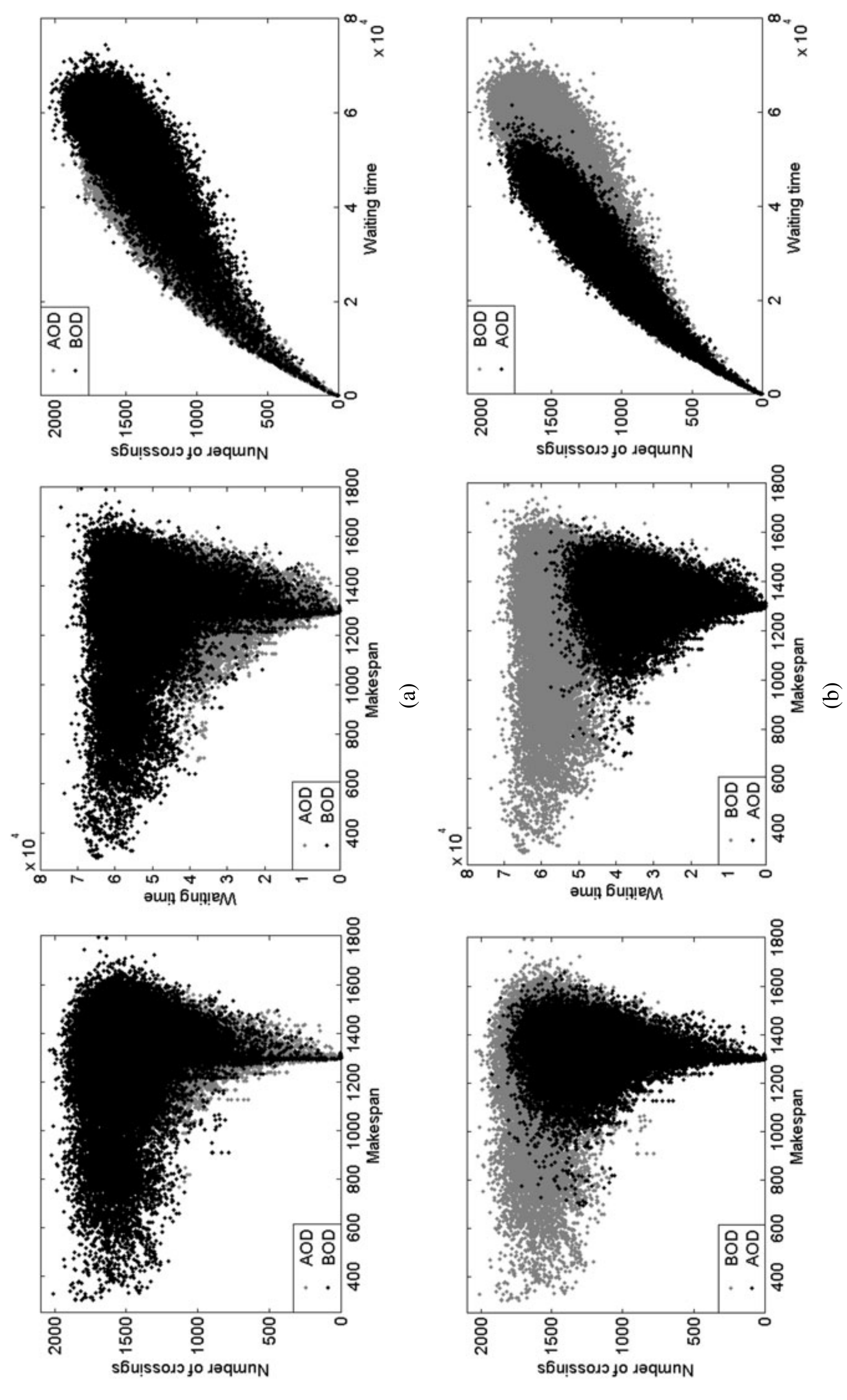


Fig. 25 Superimposing search space plots of (**a**) BOD onto AOD and (**b**) AOD onto BOD

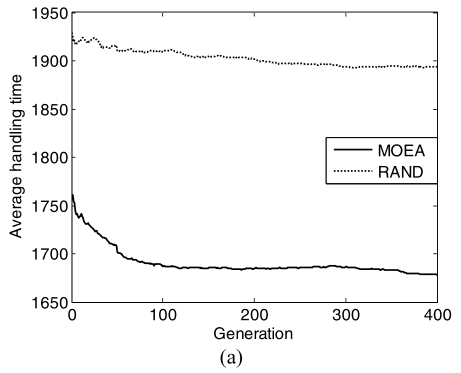


Fig. 26 Average handling time of non-dominated solutions for MOEA and RAND on (**a**) BAP5 × 100F and (**b**) BAP5 × 100L

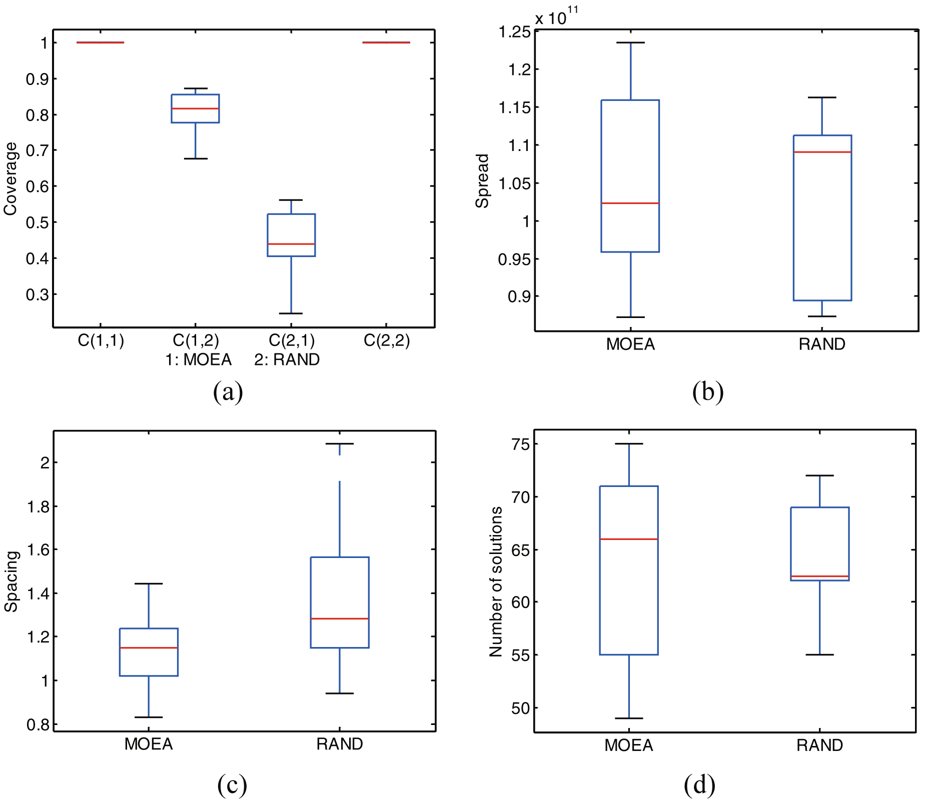


Fig. 27 (**a**) Coverage, (**b**) spread, (**c**) spacing, and (**d**) number of non-dominated solutions for MOEA and RAND on BAP5 × 100F

procedure focuses more on improving the proximity of the generated Pareto front. Unlike local search and the hybrid solution decoding scheme, it has little effect on the diversity of the obtained Pareto front.

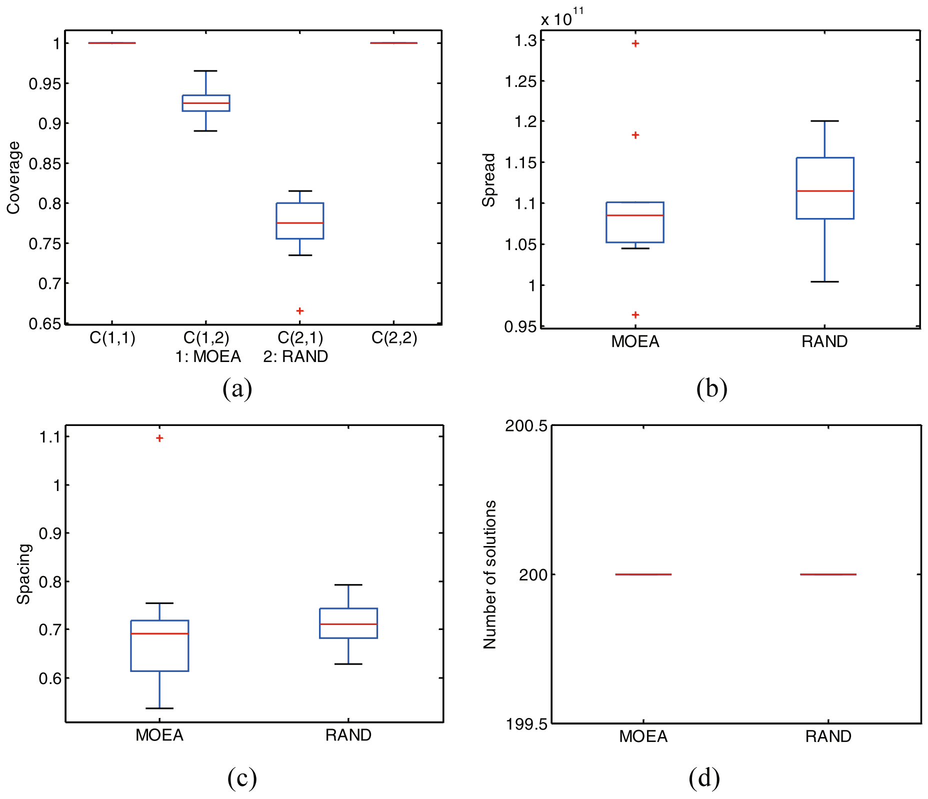


Fig. 28 (**a**) Coverage, (**b**) spread, (**c**) spacing, and (**d**) number of non-dominated solutions for MOEA and RAND on BAP5 × 100L

4.4 Performance of MOEA on other test problems

The previous sections have studied how the three primary features of the MOEA affect the quality of berth schedules. While local search reduces the number of crossings in solutions, the hybrid solution decoding scheme is able to exploit on the advantages of the berthing and assignment order decoding schemes to allow a larger search space to be explored, leading to a Pareto front that is superior in terms of proximity and diversity. Lastly, optimal berth insertion reduces the handling times of ships which in turn reduces their waiting times and the makespan of the port, further improving the proximity of the Pareto front. This section validates the optimization performance of the proposed MOEA against a simple MOEA (SMOEA) on the test problems listed in Table 3. SMOEA has the same functions as the MOEA except that it does not make use of the three proposed features. To decode solutions for fitness evaluation, SMOEA uses the berthing order decoding scheme.

The performance comparison results of the MOEA and SMOEA are shown in Figs. 29 and 30.

The comparison results show that the MOEA consistently outperforms SMOEA in terms of coverage and spread. While the MOEA is generally comparable to SMOEA in terms of spacing and superior in terms of number of non-dominated solutions generated, there are a few minor exceptions. For BAP5 × 100F, some of the simulation runs of SMOEA are able

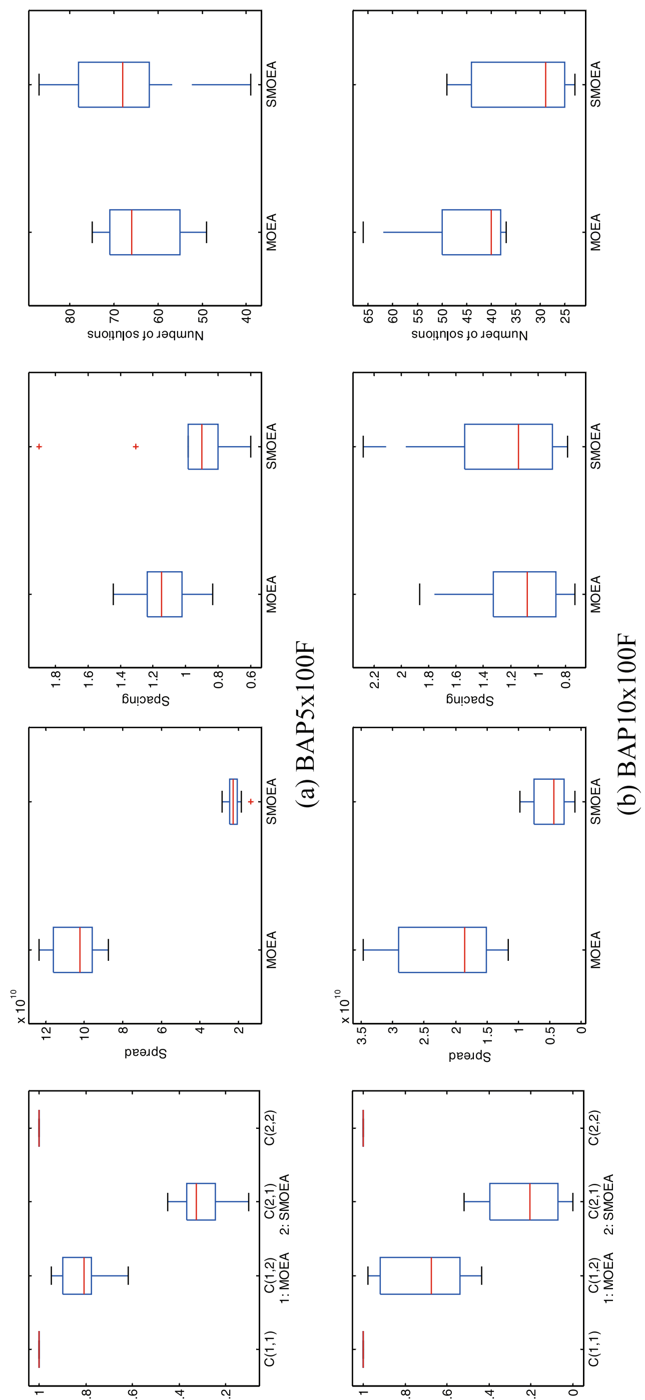
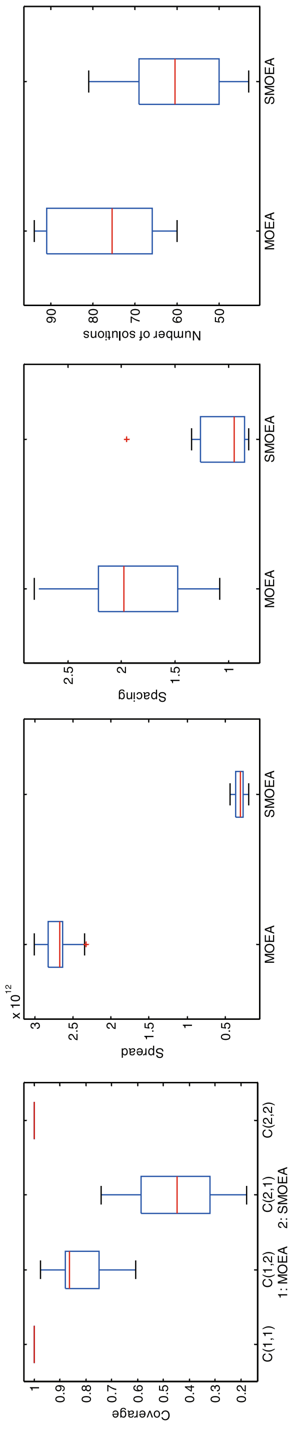
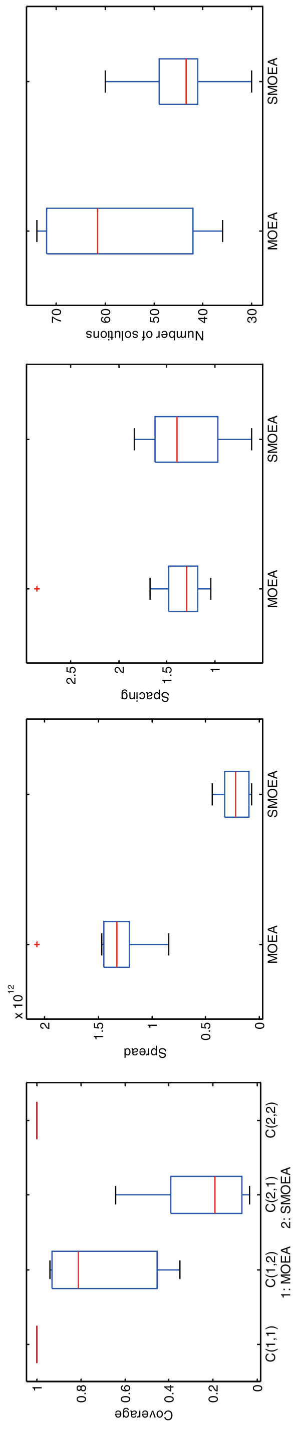


Fig. 29 Performance comparison between MOEA and SMOEA on FCFS test problems

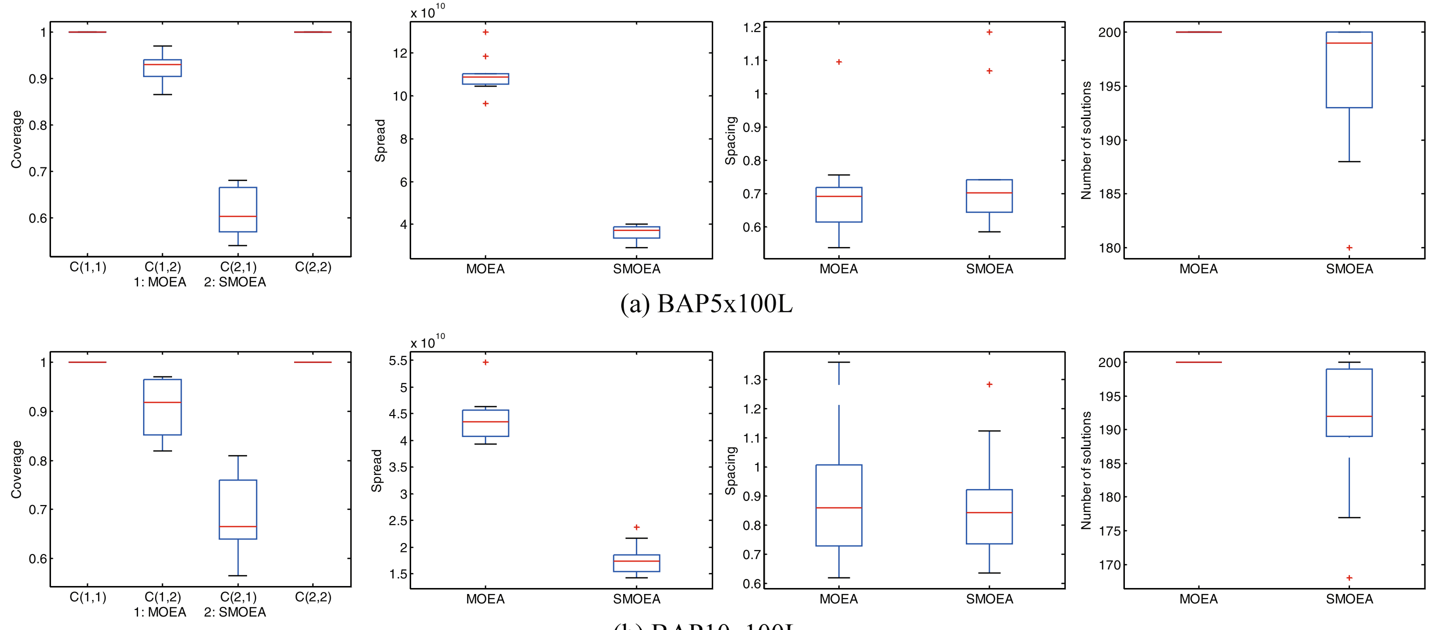


(c) BAP5x200F



(d) BAP10x200F

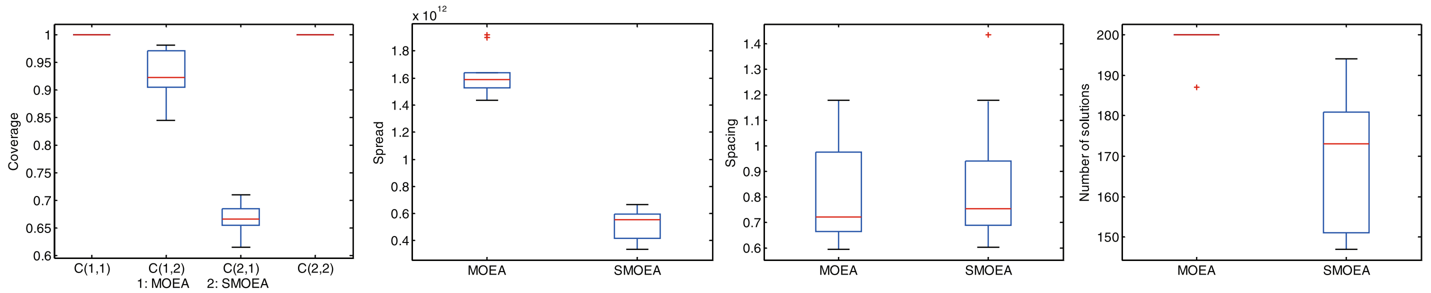
Fig. 29 (*Continued*)



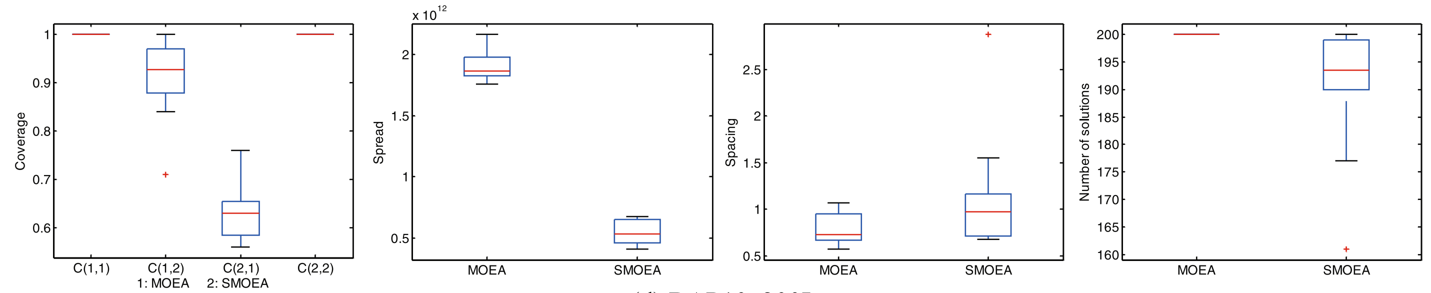
(a) BAP5x100L

(b) BAP10x100L

Fig. 30 Performance comparison between MOEA and SMOEA on LCFS test problems



(c) BAP5x200L



(d) BAP10x200L

Fig. 30 (*Continued*)

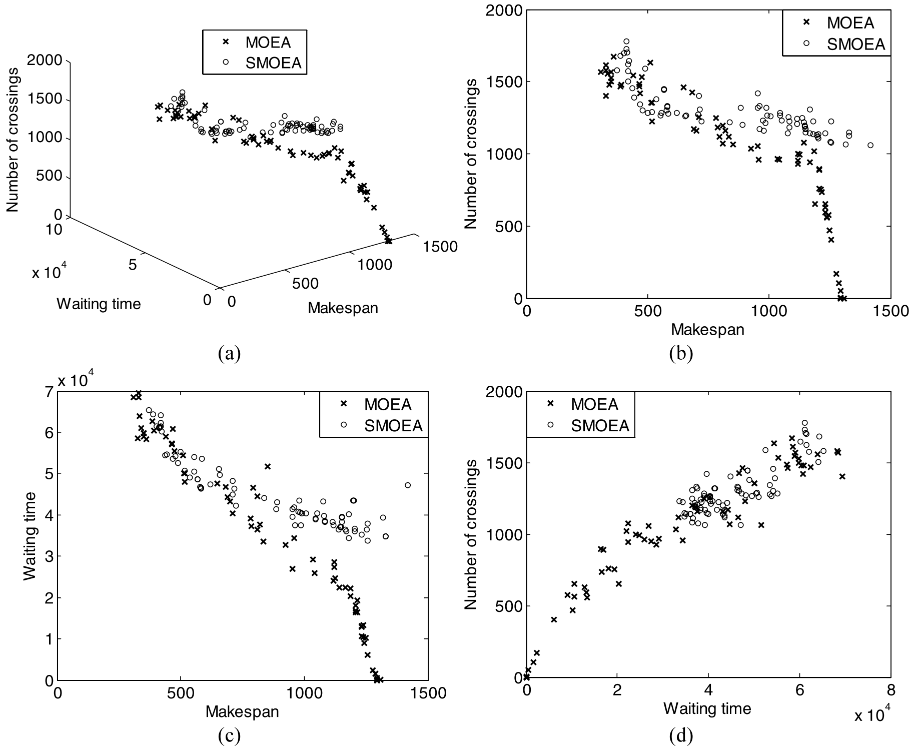


Fig. 31 Pareto fronts for a random run of the MOEA and SMOEA on BAP5 × 100F

to generate more solutions than those of the MOEA. SMOEA also performs better in terms of spacing for that test problem. A closer examination of the Pareto fronts for a random run of the MOEA and SMOEA on BAP5 × 100F in Fig. 31 reveals the superiority of the MOEA despite the slightly negative results in spacing and number of non-dominated solutions generated. Although the Pareto front obtained by SMOEA consists of more solutions (78 solutions for SMOEA compared to 67 for the MOEA), most of the solutions are inferior and dominated by the solutions in the Pareto front generated by the MOEA. From the comparison results in Figs. 29 and 30, it is evident that the three features of local search, hybrid solution decoding scheme, and optimal berth insertion play an important role in the optimization performance of the proposed MOEA.

5 Conclusions

In this paper, the berth allocation problem (BAP) has been considered as a multi-objective optimization problem that involves the minimization of the three objectives of makespan, waiting time, and degree of deviation from a predetermined priority schedule. These objectives are considered with the interests of both port and ship operators in mind. A multi-objective evolutionary algorithm (MOEA), featured with a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure, has been presented.

The proposed MOEA differs from most existing single-objective-based approaches in that it optimizes all objectives concurrently without the need of aggregating them into a compromise function. Given the intricate relationships between the three objectives that have been uncovered in this paper, the multi-objective approach appears to be the natural choice for tackling the BAP. It generates a Pareto set of berth schedules from which the port management can select a desirable solution for implementation. In addition, the effects that local search, solution decoding schemes, and optimal berth insertion have on the quality of berth schedules have been studied. It has been shown and validated that the three features play a pivotal role in the optimization performance of the MOEA.

References

Bosman, P., & Thierens, D. (2003). The balance between proximity and diversity in multi-objective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, *7*(2), 174–188.

Brown, G. G., Lawphongpanich, S., & Thurman, K. P. (1994). Optimizing ship berthing. *Naval Research Logistics*, *41*, 1–15.

Brown, G. G., Cormican, K. J., Lawphongpanich, S., & Widddis, D. B. (1997). Optimizing submarine berthing with a persistence incentive. *Navel Research Logistics*, *44*, 301–318.

Cheong, C. Y., Tan, K. C., & Veeravalli, B. (2007). Solving the exam timetabling problem via a multi-objective evolutionary algorithm—a more general approach. In *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling*, *CI-Sched 2007* (pp. 165–172), Honolulu, HI, USA, 2007.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. New York: Wiley.

Fonseca, C. M. (1995). *Multiobjective genetic algorithms with application to control engineering problems*. Ph.D. thesis, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK.

Foo, H. M. (2000). *The application of genetic algorithms to the berth allocation problem*. Master thesis, Department of Computer Science, Faculty of Science, National University of Singapore, Singapore.

Guan, Y., Xiao, W.-Q., Cheung, R. K., & Li, C.-L. (2002). A multiprocessor task scheduling model for berth allocation: heuristic and worst case analysis. *Operations Research Letters*, *30*, 343–350.

Imai, A., Nagaiwa, K., & Chan, W. T. (1997). Efficient planning of berth allocation for container terminals in Asia. *Journal of Advanced Transportation*, *31*, 75–94.

Imai, A., Nishimura, E., & Papadimitriou, S. (2001). The dynamic berth allocation problem for a container port. *Transportation Research Part B: Methodological*, *35*(4), 401–417.

Imai, A., Nishimura, E., & Papadimitriou, S. (2003). Berth allocation with service priority. *Transportation Research Part B: Methodological*, *37*(5), 437–457.

Imai, A., Sun, X., Nishimura, E., & Papadimitriou, S. (2005). Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B: Methodological*, *39*(3), 199–221.

Imai, A., Nishimura, E., Hattori, M., & Papadimitriou, S. (2007). Berth allocation at indented berths for mega-containerships. *European Journal of Operational Research*, *179*(2), 579–593.

Kim, K. H., & Moon, K. C. (2003). Berth scheduling by simulated annealing. *Transportation Research Part B: Methodological*, *37*(6), 541–560.

Lai, K. K., & Shih, K. (1992). A study of container berth allocation. *Journal of Advanced Transportation*, *26*, 45–60.

Li, C.-L., Cai, X., & Lee, C.-Y. (1998). Scheduling with multiple-job-on-one-processor pattern. *IIE Transactions*, *30*, 433–445.

Lim, A. (1998). The berth planning problem. *Operations Research Letters*, *22*(2–3), 105–110.

Nishimura, E., Imai, A., & Papadimitriou, S. (2001). Berth allocation planning in the public berth system by genetic algorithms. *European Journal of Operational Research*, *131*(2), 282–292.

Park, K. T., & Kim, K. H. (2002). Berth scheduling for container terminals by using a sub-gradient optimization technique. *Journal of the Operational Research Society*, *53*, 1054–1062.

Park, Y.-M., & Kim, K. H. (2003). A scheduling method for berth and quay cranes. *OR Spectrum*, *25*, 1–23.

Ross, P., & Corne, D. (1994). Applications of genetic algorithms. *AISB Quarterly*, *89*, 23–30.

Tan, K. C., Cheong, C. Y., & Goh, C. K. (2007). Solving multi-objective vehicle routing problem with stochastic demand via evolutionary computation. *European Journal of Operational Research*, *177*, 813–839.

Ying, Y. M. (1995). Berth allocation planning using genetic algorithms. Master thesis, Department of Civil Engineering, Faculty of Engineering, National University of Singapore, Singapore.

Zitzler, E., & Thiele, L. (1999). Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257–271.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multi-objective evolutionary algorithms: empirical results. *Evolutionary Computation*, *8*(2), 173–195.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., & Fonseca, V. G. (2003). Performance assessment of multi-objective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, *7*(2), 117–132.