Logist. Res. (2010) 2:65–78
DOI 10.1007/s12159-010-0031-8

ORIGINAL PAPER

Solving a bi-objective winner determination problem in a transportation procurement auction

Tobias Buer · Giselher Pankratz

Received: 5 February 2009 / Accepted: 14 April 2010 / Published online: 13 June 2010

Abstract This paper introduces a bi-objective winner determination problem which arises in the procurement of transportation contracts via combinatorial auctions where bundle bidding is possible. The problem is modelled as a bi-objective extension to the set covering problem. We consider both the minimisation of the total procurement costs and the maximisation of the service-quality level at which the transportation contracts are executed. Taking into account the size of real-world transport auctions, a solution method has to cope with problems of up to some hundred contracts and a few thousand bundle bids. To solve the problem, we propose a bi-objective branch-and-bound algorithm and eight variants of a multiobjective genetic algorithm. Artificial benchmark instances that comply with important economic features of the transport domain are introduced to evaluate the methods. The branch-and-bound approach is able to find the optimal trade-off solutions in reasonable time for very small instances only. The eight variants of the genetic algorithm are compared among each other by means of large instances. The best variant is also evaluated using the small instances with known optimal solutions. The results indicate that the performance largely depends on the initialisation heuristic and suggest also that a well-balanced combination of genetic operators is crucial to obtain good solutions.

T. Buer (✉) · G. Pankratz
Department of Information Systems, Faculty of Business Administration and Economics, University of Hagen, 58084 Hagen, Germany
e-mail: tobias.buer@fernuni-hagen.de

G. Pankratz
e-mail: giselher.pankratz@fernuni-hagen.de

Keywords Winner determination · Combinatorial auction · Multiobjective optimisation · Branch-and-bound · Genetic algorithm

1 Procurement of transportation contracts

Shippers, like retailers as well as industrial enterprises, often procure the transportation services they require via reverse auctions, where the objects under auction are *transportation contracts*. Usually, such contracts are designed as framework agreements lasting for a period of 1–3 years, and defining a pick-up location, a delivery location, and the type and volume of goods that are to be transported between both locations. Additionally, further details such as a contract-execution frequency, e.g., delivery twice a week, and the required quality of service, e.g., a predefined on-time delivery rate, are specified in a transportation contract. A carrier can bid for one or more contracts. In each bid, the carrier states how much he wants to be paid for accepting the specified contracts.

Transportation procurement auctions are of high economic relevance. Caplice and Sheffi [4] report on the size of real-world transportation auctions in which they were involved over a period of 5 years. According to their report, in a single transportation auction up to 470 (median 100) carriers participated, up to 5,000 (median 800) lanes were tendered, and the annual cost of transportation amounted up to US-$ 700 million (median US-$ 75 million). Elmaghraby and Keskinocak [10] present a case study of a procurement auction event in which a do-it-yourself chain operating mainly in North America procured transportation services for about a fourth of the in-bound moves to their chain stores, which corresponds to a number of over 600 lanes. In the study at hand, the terms *lane* and transportation *contract* are used interchangeably. Similarly, shippers in Europe strive to consolidate their transportation procurement activities by running European-wide tenderings. As a consequence, transportation procurement auctions in Europe have significantly increased in size and scope over the last few years which makes it difficult to manage them without the help of advanced information technology. In recent years, specialised Internet portals have emerged, which offer contractors a neutral environment for issuing their logistics contracts. Sizes of tenderings processed via such platforms reportedly scale up to several hundreds of contracts [5].

In the scenario presented here, there are a number of interesting problems on the carrier's as well as on the shipper's side. This paper focuses on the allocation problem that has to be solved by the shipper after all bids are submitted. In particular, two characteristics of the given scenario are of interest.

First, from a carrier's point of view, there are complementarities between some of the contracts. That is, the costs for executing some contracts simultaneously are lower than the sum of the costs of executing each of these contracts in isolation. The cost effect of such complementarities is also referred to as economies of scope.

Second, allocation of contracts to carriers has to be done taking into account multiple, often conflicting decision criteria. While some of the criteria (e.g., limiting the total number of carriers employed) may be naturally expressed as side constraints, other criteria should be considered explicitly as objectives. In particular, there is usually a trade-off between the classical cost-minimisation goal on the one hand and the desire for high service quality on the other. Both objectives are of almost equal importance to most shippers, cf. Caplice and Sheffi [3] and Sheffi [20].

In their recent review of the carrier selection literature, Meixell and Norbis [17] identified that the issue of economies of scope is dealt with in only a few papers and should be emphasised in future research. In order to exploit economies of scope (i.e., complementarities) between contracts in the bidding process, the use of so-called combinatorial auctions is increasingly recommended [1, 2, 20]. Combinatorial auctions allow carriers to submit bids on any subset of all tendered contracts ("bundle bids"). Through this, carriers can express their preferences more extensively than in classical auction formats. However, bundle bidding complicates the selection of winning bids. This problem is known as the winner determination problem (WDP) of combinatorial auctions. In the procurement context, the WDP is usually modelled as a variant of a set partitioning or set covering problem, both of which are NP-hard combinatorial optimisation problems. For a survey on winner determination problems, see e.g., [1].

As to the multiple-criteria property of the allocation problem, there are two ways by which most shippers solve the conflict between cost and quality goals:

One way is to restrict participation in the auction to those carriers that comply with the minimum quality standard required to meet the quality demands of any of the contracts. Thus, the service quality performance of all remaining carriers is considered equal, and the only objective is to minimise total procurement costs. Unfortunately, unless the contract requirements are fairly homogenous, this approach leads to the quality requirements of many contracts being exceeded. The second way is to take into account service-quality performance differences between carriers by applying penalties or bonuses to the bundle bid prices, depending e.g., on a carrier's service-quality in previous periods.

This paper focuses on a third alternative, which integrates quality and cost criteria by explicitly modelling the WDP as a bi-objective optimisation problem. This model extends a previous model presented in [2], which can be seen as a special case of the model presented in this paper.

Previous work does not generally focus on modelling and solving winner determination problems under explicit consideration of multiple objectives. Different kinds of winner determination problems in combinatorial auctions for transportation contracts are treated in [4, 10, 15, 20, 21]. All these studies focus on bundle bidding to exploit complementarities between contracts and consider minimisation of total procurement costs to be the only objective.

The structure of the remaining paper is as follows: Sect. 2 defines the bi-objective winner determination problem that is being studied. To solve this problem, an exact bi-objective branch-and-bound and a bi-objective genetic algorithm are introduced in Sect. 3. The algorithms are evaluated on newly generated benchmark instances in Sect. 4. Finally, Sect. 5 gives an outlook on planned future work.

2 A bi-objective winner determination problem (2WDP-SC)

The winner determination problem (WDP) of a combinatorial procurement auction with two objectives is a generalisation of the well-known set covering problem (SC). Hence the problem at hand is called 2WDP-SC. It is formulated as follows:

Given is a set of transport contracts T. Let t denote a transport contract with $t \in T$; a set of bundle bids B where a bundle bid $b \in B$ is defined as triple $b := (c, \tau, p)$. This means a carrier $c \in C$ is willing to execute the subset of transport contracts τ at a price of p. Given is furthermore a set $Q := \{q_{ct} | \forall c \in C \wedge \forall t \in T\}$ where $q_{ct} \geq 0$ indicates the quality level by which carrier c fulfils the transport contract t. Note that this is a rather expressive way to integrate quality aspects in the model. However, in practice, it may be difficult to capture all the q_{ct} values. In this case, the model also allows to represent quality levels of lower granularity depending on the granularity of the shipper's carrier assessment. At the margin, if the shipper evaluates carrier quality only on a one-value-per-carrier basis, the quality values for carrier c will be initialised as $q_{ct_1} = q_{ct_2}$ for all $t_1, t_2 \in T$.

The task is to find a set of winning bids $W \subseteq B$, such that every transport contract t is covered by at least one bid b. Furthermore, the total procurement costs, expressed in objective function f_1, are to be minimised and the total service quality, expressed in objective function f_2, is to be maximised. The 2WDP-SC is modelled as follows:

$$\min f_1(W) = \sum_{b \in W} p(b) \tag{1}$$

$$\max f_2(W) = \sum_{t \in T} \max\{q_{ct} | c \in \{c(b) | b \in W \wedge t \in \tau(b)\}\} \tag{2}$$

$$\text{s.t.} \quad \bigcup_{b \in W} \tau(b) = T. \tag{3}$$

Each transport contract t has to be chosen at least once (3). Accordingly, some contracts may be covered by two or more winning bids and therefore "paid more than once" by the shipper. Hence, preferring a set covering to a set partitioning formulation might seem at first counterintuitive. However, given the same set of bundle bids, the total cost of an optimal solution to the set covering problem never exceeds the total cost of an optimal set partitioning solution and might be even lower. Of course, a set partitioning formulation is appropriate if each carrier could be forced to submit a bundle bid on each of the $2^{|T|} - 1$ contract combinations. However, this seems unrealistic in practical scenarios due to the high number of possible combinations. For this reason, from the shipper's point of view, the set covering formulation appears more suitable. Nevertheless, if a contract is covered by more than one winning bid, there is at least one carrier who must not carry out this contract, although that carrier's bid won the auction. In the scenario at hand, this is possible, as it appears reasonable to assume free disposal [19]. In the transportation, procurement context, free disposal means that a carrier has no disadvantage if he is asked by the shipper to carry out fewer contracts than he was paid for.

The first objective function (1) minimises the total cost of the winning bids. The second objective function (2) maximises the total service-quality level of all transport contracts. Note that $\{c(b) | b \in W \wedge t \in \tau(b)\}$ is the set of carriers who have won a bid on transport contract t. Since contracts need to be executed only once but may be part of more than one winning bid, it is not appropriate to simply add up the respective qualification values of all $b \in W$. Instead, it appears reasonable to assume that the shipper will break ties in favour of the bidder who offers the highest service level for a given contract. Hence, by assumption, for each transport contract t only the maximum qualification values q_{ct} with $c \in \{c(b) | b \in W \wedge t \in \tau(b)\}$ are added up. Note that this rule might introduce an incentive for the carriers towards undesired strategic-bidding behavior. As this paper does not focus on auction-mechanism design, we leave this issue to forthcoming research.

3 Solution approaches for the bi-objective winner determination problem

To solve the 2WDP-SC, this section presents two algorithms. The first is an exact algorithm based on the idea of branch-and-bound. Taking into account the NP hardness of the bi-objective set covering problem, the non-linear objective function f_2, and the large size of real world problems, the branch-and-bound approach will probably solve only some of the relevant problems in reasonable time. Therefore, a second solution approach is presented which is an extension to a successfully applied multiobjective genetic algorithm.

Both algorithms aim to find all trade-off solutions without weighting the two objective functions. Thus, the shipper does not have to quantify his preferences, which can be challenging [20]. Both algorithms find a set of non-dominated solutions (the true Pareto set or a good approximation set, respectively). The shipper finally has to choose a solution from this set according to his subjective preferences. The latter is outside the scope of this study. For notational convenience, the 2WDP-SC is treated in the following as a pure minimisation problem, i.e., the objective function f_2 is redefined as $f_2 := (-1) \cdot f_2$ and is to be minimised.

At first, the underlying terminology is defined (cf. e.g., [25]): The set of all feasible solutions of an optimisation problem is denoted by X. A solution $x \in X$ is evaluated by a vector-valued objective function $\mathbf{f}(x) = (f_1(x), \ldots, f_m(x))$ with $\mathbf{f}(x) \in \mathbb{R}^{>}$. A solution $x^1 \in X$ dominates another solution $x^2 \in X$ (written $x^1 \prec x^2$), if and only if no component of the vector-valued objective function $\mathbf{f}(x^1)$ is larger and at least one component of $\mathbf{f}(x^1)$ is smaller than the corresponding component of $\mathbf{f}(x^2)$. A solution x^* is called *Pareto optimal* if there is no $x \in X$ that dominates x^*. The set of all Pareto optimal solutions is called *Pareto (solution) set* Ω^*. A set of solutions Ω is called an approximation of Ω^* or *(Pareto) approximation set*, if every solution in Ω is not dominated by any other solution in Ω.

3.1 A branch-and-bound algorithm based on the epsilon-constraint method

In order to solve the 2WDP-SC exactly, the Epsilon-constraint method [6, 13] is used. The idea of the Epsilon-constraint method is to optimise a single objective function, treating the other objective function as additional side constraint whose value is bounded by a particular ε. To obtain the Pareto set, a proper sequence of single objective optimisation problems has to be solved for different values of ε. Here, the 2WDP-SC is linearised by treating f_2 as side constraint. The derived single-objective minimisation problem is denoted as εWDP-SC and consists of the objective function (1) with the covering constraint (3) and the epsilon-constraint $f_2(W) < \varepsilon$.

Using a problem-independent branch-and-bound approach based on linear relaxation, though seeming natural, proved unsuitable for solving the εWDP-SC. This is due to the non-linearity of the second objective function f_2, in which for each transport contract, a max{.} term is calculated and the results are summed up over all contracts. To obtain a linear model, all max{.} terms have to be replaced by additional side constraints and additional decision variables (e.g., [22]). Compared to the $|B|$ decision variables of the non-linear εWDP-SC, the linearised variant of the model contains $|B| + |T| + |T| \cdot |B|$ decision variables. For example, even for a small problem instance with 40 bundle bids and 20 contracts, there are already 860 decision variables.

Therefore, a problem-specific branch-and-bound procedure is introduced to solve the εWDP-SC. This algorithm, referred to as εLookahead-branch-and-bound (εLBB), consists of two main components. The first component (*repeatLBBForDifferentEpsilons*, Alg. 1) iteratively selects a feasible value for ε and hands it over to the second component, the actual branch-and-bound procedure *LookaheadBB* (Alg. 2). This procedure solves the εWDP-SC to find the cost minimal solution for the given quality level ε.

Algorithm 1 repeatLBBForDifferentEpsilons

input: set of bundle bids B
$W \leftarrow LookaheadBB(B, 0)$
initialise approximation set $\Omega \leftarrow \{W\}$
$\epsilon \leftarrow f_2(W)$ // worst (highest) ε
$\epsilon^* \leftarrow f_2(B)$ // best (lowest) ε
while $\varepsilon > \varepsilon^*$ **do**
    $W \leftarrow LookaheadBB(B, \epsilon)$
    $\Omega \leftarrow \Omega \cup \{W\}$
    $\epsilon \leftarrow f_2(W)$
end while
output: Ω which is the Pareto set

Algorithm 2 LookaheadBB

input: $(b_1, \ldots, b_{\max})$, ε
$bestCost \leftarrow \infty$
$bestSolution \leftarrow \{\}$
initial problem node $PN \leftarrow \{\{\}, 1, \infty\}$
initialise *queue* and add *PN* to *queue*
while *queue* not empty **do**
    $PN \leftarrow$ problem node with minimum lower bound from *queue*
    remove *PN* from *queue*

    $contribute \leftarrow$ **false**
    if $f_1(PN.W \cup \{b_{PN.i}\}) < bestCost$ **then**
        if $\tau(b_{PN.i}) \setminus \bigcup_{b \in PN.W} \tau(b) \neq \emptyset$ **then**
            $contribute \leftarrow$ **true**
        else if $f_2(PN.W) \geq \varepsilon$ and $f_2(PN.W \cup b_{PN.i}) < f_2(PN.W)$ **then**
            $contribute \leftarrow$ **true**
        end if
    end if

    if *contribute* = **true then**
        $PN1 \leftarrow \{PN.W \cup \{b_{PN.i}\}, PN.i + 1, PN.lb\}$
        $processNode(PN1)$
    end if

    $freeBids \leftarrow \{b_i \in (b_1, \ldots, b_{max}) | i > PN.i\}$
    if $PN.W \cup freeBids$ is feasible **then**
        $PN2 \leftarrow \{PN.W, PN.i + 1, PN.lb\}$
        $processNode(PN2)$
    end if
end while
output: *bestSolution*

Alg. 1 initially determines the worst and the best possible values of f_2, which relate to the maximum and minimum ε-values, respectively (keep in mind that f_2 was redefined to a minimisation objective). On the one hand, the maximum (worst) feasible value for ε is calculated by solving the εWDP-SC using *LookaheadBB* with $\varepsilon = 0$. The obtained solution coincides with the minimal cost solution of the set covering problem. On the other hand, the minimum (best) possible value for ε, denoted as ε^*, is simply given by $f_2(B)$ (generally, B is not in the Pareto set).

After the minimum and maximum bounds for ε are known, *repeatLBBForDifferentEpsilons* triggers *LookaheadBB* to consecutively calculate the solutions in the Pareto set. Alg. 1 computes in each iteration of the while-loop one solution. The loop starts with the highest (worst) ε, calls *LookaheadBB* and then decreases ε to the f_2 value of the current Pareto solution until $\varepsilon = \varepsilon^*$. By this approach, the number of required while-iterations to find the Pareto set is minimal, i.e., the number of costly *LookaheadBB* calls is as low as possible.

The branch-and-bound procedure *LookaheadBB* (Alg. 2) solves the εWDP-SC for a given ε and the set of bundle bids B, represented as sequence $(b_i)_{b \in B}$ with $1 \leq i \leq \max$ and $\max = |B|$, by implicitly enumerating the solution space. The solution space is divided into subspaces which are represented in the branch-and-bound tree as problem nodes. Here, a problem node is a triple $PN := (W, i, lb)$ in which *PN.W* represents the current (probably incomplete) solution, i.e., the set of winning bids, *PN.i* represents the index of the bundle bid investigated in the node, and *PN.lb* is the lower bound of the current solution *PN.W* for f_1. All active problem nodes are saved in a priority *queue* according to ascending values of *PN.lb*.

The algorithm was developed according to the following main ideas:

Branching on bundle bids. Each node *PN* has two potential descendants *PN*1 and *PN*2. *PN*1 contains the current bundle bid $b_{PN.i}$ as winning bid ($b_{PN.i} \in PN1.W$), whereas *PN*2 does not ($b_{PN.i} \notin PN2.W$). Two additional rules are used to decide whether a descendant node should be generated at all:

– *PN*1 is only generated if $b_{PN.i}$ contributes to reach a feasible solution. This means that the current bundle bid $b_{PN.i}$ has to cover at least one transport contract uncovered so far, or, if the epsilon constraint is not yet met, adding $b_{PN.i}$ must reduce f_2.
– On the other hand, *PN*2 is only generated if the current winning bids *PN.W* and the remaining *free bids* jointly lead to a feasible solution with respect to both the covering and the epsilon constraints. In checking this property, the algorithm has to lookahead on future bundle bids, which led to the labelling *Lookahead* in εLBB.

Solving a relaxed problem to obtain a lower bound. For each problem node, a lower bound is calculated by solving a residual set covering problem which is defined through the remaining free bids, the transport contracts still uncovered and by dropping the integrality constraints.

LookaheadBB uses the procedure *processNode* (Alg. 3) to control how to continue processing a given PN. Provided that PN.W is feasible and a new lowest cost solution is found, the current best solution and the current best cost value are updated. Additionally, all problem nodes from the *queue* whose lower bound is less than the current best-known cost value are removed. Provided that PN.W is infeasible, a new lower bound *PN.lb* is computed. The lower bound equals $f_1(PN.W)$ plus the cost value of the optimal solution to the residual linear relaxed set covering problem. This set covering problem is defined by those contracts *not* covered by *PN.W* which have to be covered by a subset of the bundle bids given by *freeBids*.

Algorithm 3 processNode

input: problem node PN
if PN.W is feasible **then**
  if $f_1(PN.W) < bestCost$ **then**
    $bestCost \leftarrow f_1(PN.W)$;
    $bestSolution \leftarrow PN.W$
    delete all problem nodes in *queue* with lower bound $\geq bestCost$
  end if
else
  if PN.i $\leq |B|$ **then**
    PN.lb $\leftarrow f_1(PN.W)+$ cost of linear relaxed solution to the residual set covering problem.
    add PN to *queue*
  end if
end if

3.2 A genetic algorithm based on SPEA2

To heuristically solve the 2WDP-SC, a multiobjective genetic algorithm (MOGA) is applied. This approach has been proven suitable for solving hard multiobjective combinatorial optimisation problems, e.g., [8]. The proposed MOGA follows the Pareto approach and searches for a set of non-dominated solutions.

To find a Pareto approximation set, a MOGA controls a set of core heuristics. The core heuristics of a MOGA can be divided into problem-specific and problem-independent operators. For those problem-independent operators that care for the specialties of population management in the multiobjective case (fitness-assignment strategy, selection of parents and insertion of children in the population), the methods proposed by Zitzler et al. in their Strength Pareto Evolutionary Algorithm 2 (SPEA2) are applied [23, 24]. The decision to use SPEA2 relies on its competitive performance particularly for solving bi-objective combinatorial optimisation problems [24]. In addition, standard bitflip mutation and standard uniform crossover [9] have been chosen as problem-independent mutation and crossover operators, respectively.

As problem-specific operators, three core heuristics are introduced: *Simple Insert, Greedy Randomised Construction* and *Remove If Feasible*. Remove If Feasible is applied as a problem-specific mutation operator, whereas Simple Insert and Greedy Randomised Construction are both used to initialise a population as well as to repair an infeasible solution. The latter is necessary because both the uniform crossover operator and the bitflip mutation operator may end up with infeasible solutions.

Since all three problem-specific core heuristics operate on encoded individuals, the chosen encoding is presented first. A binary encoding of a solution seems suitable for set covering-based problems like the 2WDP-SC. Every gene represents a bundle bid b. If $b \in W$ the gene value is 1, and if $b \notin W$ the gene value is 0.

Simple Insert (SI) in each iteration randomly chooses a bundle bid b that contains at least one still uncovered transportation contract as a winning bid. The transport contracts τ_b in bid b are marked as covered. These steps are repeated until all contracts T are covered and SI terminates.

Greedy Randomised Construction (GRC) is inspired by the construction phase of the metaheuristic GRASP [12] and is slightly adapted for the bi-objective case (see Alg. 4). During each iteration, a winning bid is selected randomly from the restricted candidate list (RCL).

Algorithm 4 GreedyRandomisedConstruction (GRC)

```
1: input: infeasible solution W
2: while W infeasible do
3:    best bundle approximation set RCL ← {}
4:    for all b ∈ B \W
5:       if b not dominated by any b' ∈ RCL then
6:          RCL ← RCL ∪ {b}
7:       end if
8:    end for
9:    randomly chose a b from RCL
10:   W ← W ∪ {b}
11: end while
12: output: feasible solution W
```

Note that the RCL is an approximation set of best bundles, which holds only non-dominated bundles with respect to the rating function $g{:} = (g^p, g^q)$ with

$$g^p(b, W) = \begin{cases} p(b)/|\tau(b) \setminus \tau(W)| & \text{for} \quad |\tau(b) \setminus \tau(W)| > 0 \\ \infty & \text{for} \quad |\tau(b) \setminus \tau(W)| = 0 \end{cases}$$

$$g^q(b, W) = (f_2(W) - f_2(W \cup b)) / \sum_{b' \in W \cup b} |\tau(b')|.$$

Both functions assign smaller values to better bundles. While g^p rates a bundle according to the average additional costs attributed to each new (i.e., still uncovered) contract in b, g^q weights the reduction in f_2 caused by adding b to the solution by the reciprocal total number of procured contracts (in the current solution).

Remove If Feasible (RIF) randomly chooses a winning bid $b' \in W$, labels b' as visited, and removes b' from W. If after this the solution W is still feasible, then another randomly chosen winning bid (which is also labelled as visited) is removed etc. If W becomes infeasible by removing b', then b' is reinserted in W. RIF terminates if all winning bids are labelled as visited.

Via combination of the core heuristics, a set of different algorithms $\mathscr{A}$ is obtained (see Fig. 1). Each algorithm $A_i \in \mathscr{A}, i = 1\ldots8$ is denoted as a triple, e.g., A_2 is represented by (SI/BF/GRC) which reads as follows: A_2 uses SI to construct solutions, bitflip mutation (BF) as mutation operator and GRC as repair operator. Since uniform crossover is the only crossover operator, this operator is not considered as a distinctive feature in the taxonomy of Fig. 1. In order to refer to a set of algorithms, the wildcard * is used at one or more positions, e.g., (*/BF/GRC) identifies A_2 and A_6.

4 Evaluation

The εLBB and the eight MOGA variants are tested on a set of newly generated benchmark instances, which reflect some important economic features of the transportation domain. First, the generation of these instances is described. After that, the results of the εLBB and the eight MOGA variants are presented.

4.1 Generating test instances

To the best of our knowledge, no benchmark instances exist for a multiobjective WDP like the proposed 2WDP-SC. However, there are several approaches for generating problem instances for single-objective winner determination problems with various economical backgrounds, e.g., the combinatorial auction test suite "CATS" of Leyton-Brown and Shoham [16] or the bidgraph algorithm introduced by Hudson and Sandholm [14]. To generate test instances for the 2WDP-SC, some ideas of the literature are extended to incorporate features specific to the procurement of transportation contracts.

As this investigation does not address any game theoretical issues like strategic bidding and incentive compatibility, it is assumed that carriers reveal their true preferences. Thus, the terms "price" and "cost valuation" of a contract combination can be used synonymously. General requirements of artificial instances for combinatorial auctions are stated by Leyton-Brown and Shoham. Both postulations seem self-evident but have not always be accounted for in the past [16]:

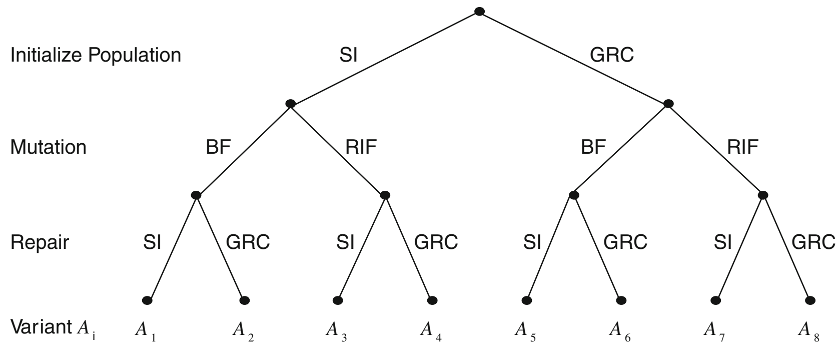

Fig. 1 Eight possible combinations of core heuristics to form an algorithm A_i

- Some combinations of contracts are more frequently bid on than other combinations. This is due to usually different synergies between contracts.
- The charged price of a bundle bid depends on the contracts in this bundle bid. Simple random prices, e.g., drawn from [0,1], are unrealistic and can lead to computationally easy instances.

Furthermore, it seems reasonable to demand that the following additional requirements specific to transportation procurement auctions are met:

- All submitted bids are binding and exhibit additive valuations (OR-bids, cf. [18]). Hence, a carrier is supposed to be able to execute any combination of his submitted bids at expenses which do not exceed the sum of the corresponding bid prices. Extra costs do not arise. Due to the medium-term contract period of 1–3 years in the scenario at hand, capacity adjustments are possible in order to avoid capacity bottlenecks. Furthermore, the carrier has the opportunity to resell some contracts to other carriers who guarantee the same quality of service.
- From the previous assumption, it follows that a rational carrier c does only bid on combinations of contracts that exhibit *strictly subadditive cost valuations*. The cost valuation of a set of contracts τ is called strictly subadditive, if for each partition $\mathscr{T}$ of the set τ, the cost valuation of τ is strictly lower than the sum of the cost valuations of all parts of the respective set partition. Formally, the carrier-specific set Π^c of all strict subadditive bids can be defined as expressed in the following formula, in which $P(\tau)$ denotes all set partitions of τ and $\mathbf{P}(\tau)$ denotes the power set of τ:

$$\Pi^c = \left\{ \tau \subseteq T^c | \forall \mathscr{T} \in P(\tau) : p^c(\tau) < \sum_{\tau' \in \mathscr{T}} p^c(\tau') \right\},$$

$$\text{with } P(\tau) = \left\{ \mathscr{T} \subset \mathbf{P}(\tau) | \bigcup_{\tau' \in \mathscr{T}} \tau' = \tau \wedge \bigcap_{\tau' \in \mathscr{T}} \tau' = \emptyset \right\}$$

Strict subadditivity in terms of cost is due to synergies between contracts. Bids composed of contracts which exhibit strict subadditive cost valuations are referred to as *essential* bids. Since all submitted bids are supposed to be OR-bids, any non-essential bid could always be replaced by an equivalent combination of two or more essential bids. Therefore, bidding on non-essential bids is redundant.

- The 2WDP-SC was modelled as a set covering problem, as it appeared reasonable to assume *free disposal*. Free disposal means that the price charged by carrier c for any subset of a set of contracts τ is not greater than the price carrier c would charge for τ. Formally, this is expressed in the following formula, in which B^c denotes the set of bundle bids submitted by carrier c:

$$p(b') \leq p(b) \mid \forall \tau(b') \subseteq \tau(b) \wedge b, b' \in B^c.$$

To be an instance suited to the 2WDP-SC, the bundle bids of each carrier should also feature the free disposal property.
- Finally, it is assumed that the carrier-specific costs of a transport contract depend on both the contract's resource requirements and the service-quality level at which the carrier is able to perform the contract.

The bids are generated using Algorithm 5, which takes four values as input: the number *nBids* of bids to be generated, the sets C and T that represent carriers and transport contracts, respectively, and the density ρ of the synergy matrix. The synergy matrix consists of binary values, which indicate the pairwise synergies between contracts. Synergies between contracts imply that the respective contract combination is cost subadditive. A higher density tends to result in more and larger contract combinations a carrier has to consider.

First of all, *BidGeneration* (Alg. 5) initialises some variables. For each carrier, a subset of contracts T^c is determined as the set of contracts that the carrier is supposed to be willing to bid for. While it is not necessary that all T^c are disjoint, they must jointly cover all contracts in T. After that, the following steps are performed for each carrier. First, the carrier-specific synergy matrix is randomly filled according to density ρ. The service-quality q_{ct} at which carrier c is able to execute contract t is chosen randomly from the integer values one to five, with higher values indicating a higher service level. Furthermore, to each contract, a resource demand r_{ct} is assigned. This is an abstract indicator for the resources required by a carrier c to carry out contract t. The resource demand of a given contract may vary from carrier to carrier, as carriers might have, e.g., different locations of their depots, different types of vehicles or existing transportation commitments which influence the required resources. The values r_{ct} are chosen randomly between 0.1 and 0.5.

Algorithm 5 BidGeneration

```
1: input: nBids, density of synergy matrix ρ, T, C
2: ∀ c ∈ C: randomly select relevant contracts T^c ⊂ T, such that ⋃_{c∈C} T^c = T
3: for all carriers c ∈ C do
4:     ∀ i, j ∈ T^c: set s^c_ij ← 1 with probability ρ, indicating that
       between contracts i and j exist synergies
5:     ∀ t ∈ T^c: randomly set contract quality q_ct ∈ {1, 2, 3, 4, 5}
6:     ∀ t ∈ T^c: randomly set resource demand r_ct ∈ [0.1, 0.5]
7:     determine essential contract combinations Π^c
8:     subadditiveBidGraphAlgorithm(Π^c) to calculate prices p(τ)
       for each τ ∈ Π^c.
9: end for
10: ∀c ∈ C : B^c ← select(Π^c)
11: output: all carrier bids B = ⋃_{c∈C} B^c
```

To obtain the set of essential contract combinations in line 7, assume for each carrier c a synergy graph $SG^c = (T^c, E^c)$. Let the vertices be the contracts T^c carrier c is interested in. If two contracts $i, j \in T^c$ feature synergies, that is $s^c_{ij} = 1$, then both contracts are connected via an edge, that is $E^c = \left\{(i,j) | s^c_{ij} = 1 \wedge i,j \in T^c\right\}$. It is assumed that any number of contracts can be combined in a single bid, as long as the sum of the corresponding resource demands does not exceed a maximum total resource demand of 1. This capacity limit is motivated by the fact that unfolding of complementarities generally is subject to resource limitations. For example, contracts often feature synergies if they are carried out conjointly in the same tour, which, however, is subject to vehicle capacity restrictions. The resource demand of each contract $t \in T^c$ is given by r_{ct}. Then, the set of feasible essential combinations of contracts equals the set of all possible induced subgraphs of SG^c with $\sum_t r_{ct} \leq 1$.

In the next step, a price for each combination of contracts is determined using the *SubadditiveBidGraph* algorithm, which is explained below. After that, the *select* operator choses among all feasible contract combinations those combinations on which each carrier is supposed to place his bids. Therefore, all contract combinations in Π are rated according to two criteria: average cost per contract $p(b)/|\tau(b)|$ and average quality per contract $\sum_{t \in \tau(b)} q_{c(b)t}/|\tau(b)|$. Then, the best contract combinations with respect to these criteria are selected according to the dominance concept. In doing so, *select* makes sure that on the one hand, the total number of bids submitted by all bidders is *nBids*, and on the other hand, each $t \in T$ is covered by at least one bundle bid to obtain a solvable instance.

The *SubadditiveBidGraph* algorithm (cf. Alg. 6) is applied to determine prices for the essential contract combinations, which comply with the assumptions of free disposal and strict subadditivity. The algorithm is based on the approach of Hudson and Sandholm [14], which generates bids with free disposal. This approach is extended, such that all generated bids also show strictly subadditive cost valuations.

Algorithm 6 SubadditiveBidGraph

```
input: set of essential contract combinations Π^c, carrier c
A^sup ← {(i,j) | i,j ∈ Π^c and i ⊂ j}
A^sub ← {(i,j) | i,j ∈ Π^c and i ⊃ j}
initialise bidgraph BG ← (Π^c, A^sup, A^sub)
∀τ ∈ Π^c with |τ| = 1 :
    UB(τ) ← LB(τ) ← p(τ) ← RandomBasePrice(τ, c)
initialise lower bounds
    ∀τ ∈ Π^c with |τ| = 1 : UpdateLowerBounds(BG, τ)
initialise upper bounds
    ∀τ ∈ Π^c with |τ| > 1 : UB(τ) = Σ_{t∈τ} p(t)
k ← 2
while k ≤ |Π| do
    for all τ ∈ {τ ∈ Π | |τ| = k ∧ LB(τ) ≠ UB(τ)} do
        set price randomly LB(τ)←UB(τ)←p(τ)∈]LB(τ),UB(τ)[
        UpdateLowerBounds(BG, τ)
        UpdateUpperBounds(BG, τ)
    end for
    k ← k + 1
end while
output: prices p(τ) for each τ ∈ Π consistent to the free
    disposal and the subadditivity assumption
```

The idea of the original bidgraph algorithm as proposed by Hudson and Sandholm is to define lower bounds $LB(\tau)$ and upper bounds $UB(\tau)$ for each considered contract combination τ such that free disposal holds. Then the procedure successively draws a price for each contract combination between its lower and upper bounds; this price is propagated through the bidgraph to sharpen the lower and upper bounds of the remaining contract combinations.

In order to extend this approach to support contract combinations that exhibit both free disposal and strictly subadditive cost valuations, the bidgraph is initialised as follows: The vertices of the bidgraph BG represent all essential contract combinations Π. There are two sets of arcs, A^{sup} and A^{sub}. The arcs in A^{sup} indicate a superset relation, i.e., an arc (i, j) from vertex i to j means that the contracts in j are a superset of the contracts in i. Similarly, the arcs in A^{sub} represent all subset relationships.

In line 5 through 8 of Alg. 6, the lower and upper bounds of all k-combinations of contracts are initialised. For a given $k \in \mathbb{N}$, let the set of all k-combinations of contracts be defined as $\{\tau \in \Pi : |\tau| = k\}$. The lower bounds *LB* for all single contracts ($k = 1$) are initialised by Algorithm 7. The price $p(\{t\})$ of a single contract t is a random variable that is normal distributed with mean μ and variance σ^2. The values of $p(\{t\})$ are forced into the interval [*minPrice*, *maxPrice*] with *minPrice* = 0.5 and *maxPrice* = 1.5. As stated above, higher resource requirements and a higher service level should tend to result in a higher price. Thus, μ depends on the resource demand r_{ct} and the service quality q_{ct} of contract t. The variance σ^2 is set to 1.0.

After *RandomBasePrice* (Alg. 7) has initialised the *LB* of all 1-combinations, Alg. 8 recursively propagates these prices through the bidgraph and updates the lower bounds of all superset contract combinations if necessary. By now, the upper bounds for the k-combinations, $k > 1$, can be calculated as the sum of the prices of all respective 1-combination contracts.

Algorithm 7 RandomBasePrice

```
input: single-contract set {t}, carrier c
minPrice ← 0.5
maxPrice ← 1.5
resources_multiplier ← r_ct/0.3      //expected mean of r_ct (Alg. 5)
qualification_multiplier ← q_ct/3    //expected mean of q_ct (Alg. 5)
μ ← 1.0+ resources_multiplier · qualification_multiplier
σ² ← 1.0
p({t}) ← normal distributed random variable with mean μ and
         variance σ²
if p({t}) > maxPrice OR p({t}) < minPrice then
    RandomBasePrice({t}, c)
end if
output: p({t})
```

Algorithm 8 UpdateLowerBounds

```
input: BG, τ
for all τ' ∈ BG.Π|(τ, τ') ∈ BG.A^sup do
    if LB(τ') < p(τ) then
        LB(τ') ← p(τ)
        UpdateLowerBounds(BG, τ')
    end if
end for
```

To ensure strictly subadditive valuations, the while-loop of Alg. 6 sets the bid prices for all k-combinations in the order of non-decreasing k, starting with $k = 2$. For all k-combinations with $LB(\tau) \neq UB(\tau)$, a price is drawn randomly between $LB(\tau)$ and $UB(\tau)$ and propagated through the bidgraph to adjust the lower and upper bounds of the other contract combinations.

In doing so, it must be assured that the upper bound never exceeds the costs of any partition of τ since this may lead to inconsistencies with respect to the subadditivity requirement. Therefore, Alg. 9 solves a set partitioning problem to optimality. The instance of the set partitioning problem is given by the sets $\{j \mid (\tau, j) \in A^{sub}\}$ and the associated costs $UB(j)$.

Algorithm 9 UpdateUpperBounds

```
input: BG, τ
for all τ' ∈ BG.Π|(τ, τ') ∈ BG.A^sup do
    p* ← price of optimal set partitioning solution to {τ' | (τ, τ')
         ∈ BG.A^sub} and associated UB(τ')
    if p* < UB(τ') then
        UB(τ') ← p*
        UpdateUpperBounds(BG, τ')
    end if
end for
```

The *BidGraphAlgorithm* continues until the prices of all essential bids are set. After that, the *select*-Operator of Alg. 5 is applied as described above. The procedure keeps generating bids for all carriers, until the test instance is complete.

4.2 Measuring the quality of an approximation set

To compare the performance of single-objective heuristics in terms of achieved solution quality, a major step is to compare the objective function values of the best found solutions, respectively. The matter is more complicated in the bi-objective case, as approximation sets have to be compared. Often there are no clear dominance relations between the solutions of different approximation sets, see e.g., Fig. 2a. Therefore, various indicators to measure the quality of approximation sets are discussed in the literature, cf. [25] for a detailed discussion of the state of the art.

To evaluate the solution quality of an approximation set, the popular hypervolume indicator I_{HV} is used [23]. I_{HV} measures the dominated subspace of an approximation set, bounded by a reference point *RP*. *RP* must be chosen such that it is dominated by all solutions of the approximation set. Furthermore, the reference point has to be identical for all compared heuristics on the same problem instance. Here, for each instance, *RP* is defined as $\left(f_1^{max}; f_2^{max}\right) = (f_1(B); 0)$, respectively.

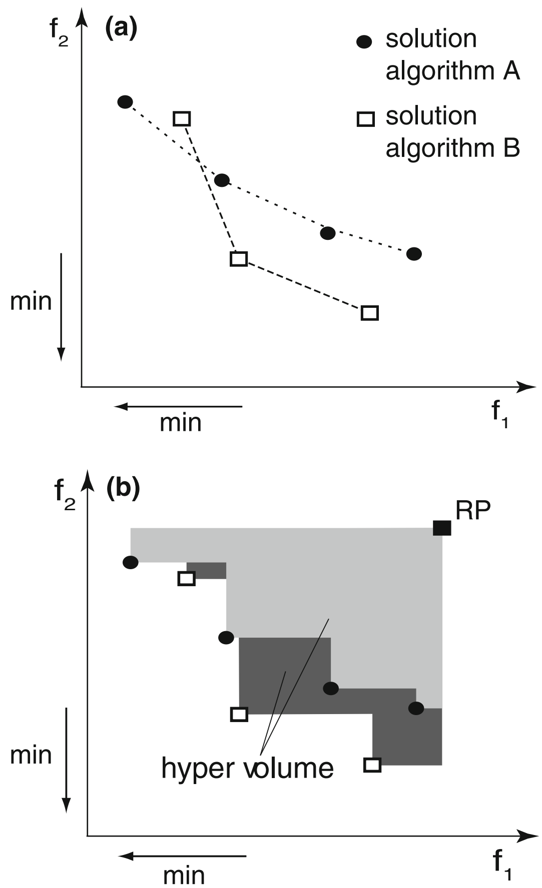

Fig. 2 Illustration of hypervolume indicator I_{HV}. **a** Solutions of two approximation sets found by two algorithms A and B. **b** The *shaded areas* of each algorithm depict the dominated subspace, respectively. Note that the *light-shaded area* is overlapping the *dark-shaded area* in part. The volume of the *dark-shaded area* is greater than the volume of the *light-shaded area*, therefore algorithm B is considered better than algorithm A

Furthermore, the objective values of all solutions are normalised according to $f_i = \left(\overline{f}_i - f_i^{\min}\right) / \left(f_i^{\max} - f_i^{\min}\right)$ with $i = 1, 2, f_1^{\max} = f_1(B), f_2^{\min} = f_2(B) - 1, f_2^{\max} = 0$. Thus, values of I_{HV} range from zero to one, and larger values indicate better approximation sets. However, as *RP* can be chosen freely to a large degree, I_{HV} is an interval-based measure. Therefore, the quality gap between algorithms can only be expressed via absolute differences of I_{HV}, but not via percentage ratios of I_{HV}.

4.3 Evaluation of the εLookahead–branch-and-bound

The εLBB was implemented in Java 6. A floating point precision of ten digits is used. The lower bounds are calculated by Dantzig's Simplex Algorithm in the implementation of the Apache Commons Math Library (version 2.1). The algorithm was tested on an Intel Pentium 4 (2.0 GHz) with 500MB RAM available to the Java Virtual Machine.

Preliminary testing gave evidence that computation times of εLBB rapidly increase with the number of bundle bids. Even moderate problem sizes caused the εLBB to run several hours before terminating. Therefore, a set of eight rather small test instances was generated according to Sect. 4.1 in order to evaluate εLBB. The instances vary only in the number of bundle bids (up to 80) and in the number of transport contracts (up to 40). The number of participating carriers and the density of the synergy matrix are held constant with values of 10 and 50%, respectively.

The results of these instances are reported in Table 4 in Sect. 4.4.2. The table shows the number of solutions in the Pareto set and the required runtime in seconds. In addition, the table contains results from the MOGA, which will be discussed in more detail in Sect. 4.4.2.

The findings demonstrate that εLBB is suited to solve small instances with up to 60 bundle bids in less than an hour. For solving problem instances with 80 bundle bids, εLBB consumes several hours of runtime. The test of the instance with 80 bundle bids and 40 contracts was aborted after a runtime of 24 h. These results strongly suggest that exact approaches like the εLBB are inappropriate as a solution approach for practical procurement scenarios which easily reach problem sizes of several hundreds of bundle bids. Nevertheless, for small instances, the optimal solutions obtained by the εLBB provide a valuable benchmark for evaluating the quality of heuristic approaches like the MOGA (cf. Sect. 4.4.2).

4.4 Evaluation of the genetic algorithm

The eight genetic algorithms A_1 to A_8, (cf. 2) were tested on the same hardware platform as the εLBB (Pentium 4, 2.0 GHz, 500 MB Ram available to the Java Virtual Machine). The problem-specific heuristics were coded in Java 6; for the problem-independent parts the SPEA2 distribution coded in C was used [11].

For the evaluation of the genetic algorithms, two data sets were considered.

On the one hand, problem instances of practical size as reported in Sect. 1 were generated. These instances are referred to as *large instances*. The instances vary in the number of bids (500–2,000), the number of contracts (125–500) and the number of carriers (25–100). In addition, the density ρ of the synergy matrix was varied (25–75%). With respect to the observation that auctions with fewer transport contracts usually tend to attract fewer bidders, it appeared reasonable to restrict the combinations of instance parameter values to those shown in Table 2. Since for the large instances, absolute benchmarks in the form of optimal solutions are not available, the relative performance of the eight MOGA variants on these instances is compared instead. The results of these tests are discussed in Sect. 4.4.1.

On the other hand, the small instances described in Sect. 4.3 were also used to evaluate the eight genetic algorithms. The results for these instances are compared to those of the exact εLBB algorithm in Sect. 4.4.2.

4.4.1 Results and discussion for large instances

In this section, the relative performance of the eight MOGA variants is evaluated using the large problem instances. The parameter values of the genetic algorithms were derived from some preliminary testing. Two to five alternative values for each parameter were tested on three randomly selected instances. The values that gave the best results in manageable time are those presented in Table 1. The same values were applied to all MOGA variants and were kept constant through all experiments.

The results for the hypervolume indicator are presented in Table 2. The last column indicates the I_{HV} value of the reference approximation set $\overline{\Omega} = \bigcup_{A \in \mathscr{A}} \Omega^A$.

The results in Tables 2 and 3 were statistically evaluated with the Kruskal–Wallis and the Mann–Whitney rank sum test. All statistical conclusions are stated at a significance level of 5%. With respect to the given test instances, the compared heuristics and the applied quality indicator, the following conclusions may be drawn.

- The probability distributions of the I_{HV} values of the eight algorithms differ significantly. The ranks given in Table 2 are derived by a systematic pairwise comparison of the hypervolume values using the Kruskal–Wallis rank test.
- A_8 performs very well, as could be expected, since it incorporates three problem-specific heuristics. According to the Kruskal–Wallis rank test, taking into account all 240 outcomes, A_8 dominates all other algorithms but A_7. A_8 computes the best results for 25 out of 30 test instances, followed by A_7 which achieves the highest value 5 times, and A_5 which scores 4 times the best value.
- The variants A_1, A_2, A_3, A_4 that belong to the class (SI/*/*) never achieve a best value in any one of the instances (cf. Table 2).

Table 1 Chosen parameter values for the test

Parameter	Value
Size of population	50 individuals
Uniform crossover-probability	15%
Bit-exchange-probability in uniform-crossover	50%
Mutation-probability	100%
Bitflip-probability	10%
Runtime	300 s
No. of parents μ for creating λ offspring	4
No. of offspring λ generated by μ parents	4

- The impression that a weak initial population significantly compromises final solution quality even if more elaborate mutation and repair operators are used intensifies by considering test no. 1 in Table 3. The approximation sets derived by the class of algorithms which use GRC as initialisation heuristic clearly outperform the class of algorithms which use SI as initialisation heuristic. This is true even on a significance level of 0.0001.
- From the fact that the overall performance strongly depends on the initialisation heuristic, one can assume that any effort invested here will be rewarded.
- Tests 2 and 3 give no hints that the more intelligent operators RIF and GRC (applied in the repair phase) promise better results than BF and SI in the general case. However, the performance of RIF significantly improves if it is applied to an intelligently initialised population (test 5, test 4).
- Tests 6 and 7 give evidence that the mutation operators BF and RIF do not show different behaviour, even if the repair operator is changed. However, if RIF is applied successfully to an individual, then there is no need to apply any repair operator, as the operator leaves the individual feasible by definition.
- Interestingly, an influence of the repair heuristic on the performance of all algorithms is not observable (test 8–11). This result gets emphasised as we could not prove a significant performance advantage of A_8 over A_7 (both differ only in the applied repair operator). This followed from the Kruskal–Wallis Test, which takes into account all 240 observations (30 instances, 8 algorithms). However, statistics paint a different picture if only the 60 observations resulting from A_7 and A_8 are compared with a signed rank test. Then, A_8 clearly outperforms A_7. Hence, in well-balanced algorithms, the repair operator may be of importance.

4.4.2 Results and discussion for small instances

For the set of small instances, the solutions found heuristically by the GA are now compared to the Pareto optimal solutions found by εLBB. This is done to gain more insights into the performance of the MOGA, especially whether the MOGA is capable of finding optimal solutions and how close the approximate solutions are to the Pareto front.

The seven small instances that could be solved by εLBB (cf. Sect. 4.3) were computed by all eight variants of the GA. As before, the computing time was fixed to 5 min. In accordance with the results for the large instances, the variant A_8 performed best, i.e., in all seven instances it reached the best hypervolume value. For this reason, Table 4 compares only the results of A_8 to the Pareto optimal solutions.

Table 2 Comparison of I_{HV} for eight MOGA variants applied to the set of 30 large test instances (specified by columns 1–4)

|B|	|T|	|C|	ρ	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	$\overline{\Omega}$
500	125	25	25	0.8473	0.8476	0.8482	0.8252	0.8663	0.8663	0.8878	**0.8914**	0.8914
			50	0.8623	0.8627	0.8624	0.8605	0.8827	0.8827	0.9028	**0.9038**	0.9038
			75	0.8614	0.8612	0.8693	0.8667	0.8759	0.8759	0.8937	**0.8983**	0.8983
1,000	125	25	25	0.9167	0.9170	0.8943	0.8509	0.9371	0.9371	0.9466	**0.9479**	0.9480
			50	0.9223	0.9220	0.8754	0.8652	0.9499	0.9499	**0.9523**	0.9508	0.9523
			75	0.9341	0.9340	0.9233	0.9117	0.9490	0.9490	0.9533	**0.9535**	0.9536
	250	25	25	0.8623	0.8623	0.8725	0.8725	0.8818	0.8818	0.8961	**0.9021**	0.9021
			50	0.8627	0.8625	0.8647	0.8579	0.8720	0.8720	0.8948	**0.9001**	0.9001
			75	0.8555	0.8553	0.8573	0.8648	0.8736	0.8736	0.8953	**0.8961**	0.8967
		50	25	0.8482	0.8488	0.8235	0.8199	0.8864	0.8865	0.8924	**0.8927**	0.8974
			50	0.8498	0.8482	0.8417	0.8357	0.8811	0.8811	0.8935	**0.8943**	0.8943
			75	0.8500	0.8497	0.8431	0.8407	0.8800	0.8800	**0.8937**	**0.8937**	0.8937
2,000	125	25	25	0.9553	0.9547	0.8843	0.8812	**0.9748**	**0.9748**	0.9720	0.9720	0.9772
			50	0.9586	0.9584	0.8944	0.8751	0.9778	0.9778	0.9785	**0.9786**	0.9786
			75	0.9615	0.9614	0.9277	0.9213	**0.9757**	**0.9757**	0.9745	0.9746	0.9764
	250	25	25	0.9268	0.9267	0.9150	0.9052	0.9516	0.9516	**0.9531**	**0.9531**	0.9531
			50	0.9282	0.9277	0.9148	0.9130	0.9471	0.9471	0.9522	**0.9532**	0.9532
			75	0.9261	0.9262	0.9317	0.9315	0.9440	0.9440	0.9486	**0.9510**	0.9510
		50	25	0.9150	0.9148	0.8387	0.8229	**0.9472**	0.9469	0.9331	0.9337	0.9498
			50	0.9228	0.9223	0.8775	0.8780	0.9494	0.9494	**0.9530**	**0.9530**	0.9546
			75	0.9221	0.9222	0.8993	0.8983	0.9471	0.9471	0.9505	**0.9508**	0.9516
	500	25	25	0.8700	0.8700	0.8880	0.8972	0.8911	0.8911	0.8988	**0.9022**	0.9022
			50	0.8601	0.8601	0.8785	0.8863	0.8837	0.8837	0.8933	**0.8942**	0.8942
			75	0.8579	0.8579	0.8807	0.8848	0.8777	0.8777	0.8885	**0.8944**	0.8944
		50	25	0.8503	0.8499	0.8490	0.8510	0.8810	0.8810	0.8901	**0.8902**	0.8907
			50	0.8605	0.8600	0.8587	0.8667	0.8826	0.8826	0.8938	**0.8947**	0.8947
			75	0.8532	0.8529	0.8694	0.8666	0.8776	0.8776	0.8886	**0.8887**	0.8894
		100	25	0.8367	0.8347	0.8263	0.8165	**0.8726**	0.8715	0.8708	0.8708	0.8809
			50	0.8433	0.8416	0.8254	0.8305	0.8798	0.8770	0.8825	**0.8827**	0.8900
			75	0.8468	0.8448	0.8465	0.8370	0.8803	0.8803	**0.8939**	**0.8939**	0.8939
Rank				5.5	5.5	7.5	7.5	3.5	3.5	1.5	1.5	
Mean				0.8848	0.8848	0.8699	0.8676	0.9081	0.9079	0.9160	0.9166	
Standard dev.				0.0405	0.0405	0.0308	0.0310	0.0378	0.0378	0.0311	0.0311	

All I_{HV} values were obtained in a single run for each of the eight MOGA variants A_1 to A_8. All runs were terminated after 5 min (300 s)

In Table 4, column ΔI_{HV} shows the difference of the hypervolume attained $I_{HV}(\varepsilon\text{LBB}) - I_{HV}(A_8)$ by the two algorithms. The third column to the right states for each instance the number of solutions in the Pareto set. In addition, the second column to the right specifies the number of solutions found by A_8 which are Pareto optimal, i.e., which are members of the Pareto solution set derived by εLBB.

The GA variant A_8 is able to find optimal solutions for six out of seven instances. No optimal solution was found for the 60 bundle-bids/40 contracts instance. For the smallest instance (for which it is trivial to generate all possible solutions), all solutions of the Pareto set are found and ΔI_{HV} equals zero.

Note that in general, a higher number optimal solutions found by the GA does not necessarily imply that the corresponding approximation set is closer to the true Pareto set. In particular, an approximation set that does not contain any optimal solution still may be quite close to the Pareto frontier. For example, consider the 60 bundle, bids/40 contracts instance for which A_8 does not find any optimal solution. Nevertheless, ΔI_{HV} indicates that A_8 obtains a good approximation of the true Pareto frontier. The Pareto frontier and the approximation frontier attained by A_8 for this instance are simultaneously visualised in Fig. 3. Though being close to the optimal points, the solutions of A_8 appear slightly shifted to the right.

Table 3 Statistical comparison of selected sets of algorithms

No.	A_i vs. A_j	H_0	$\hat{\alpha}$ (%)
1	(*GRC/*/**) *vs.* (*SI/*/**)	reject	0.01
2	(*/*RIF*/*) *vs.* (*/*BF*/*)	–	73.85
3	(*/*/*GRC*) *vs.* (*/*/*SI*)	–	91.41
4	(*SI/BF/**) *vs.* (*SI/RIF/**)	–	16.03
5	(*GRC/RIF/**) *vs.* (*GRC/BF/**)	reject	0.01
6	(*/*RIF/SI*) *vs.* (*/*BF/SI*)	–	75.48
7	(*/*RIF/GRC*) *vs.* (*/*BF/GRC*)	–	67.65
8	(*SI/*/GRC*) *vs.* (*SI/*/SI*))	–	88.52
9	(*GRC/*/GRC*) *vs.* (*GRC/*/SI*)	–	80.11
10	(*/*BF/GRC*) *vs.* (*/*BF/SI*)	–	93.10
11	(*/*RIF/SI*) *vs.* (*/*RIF/GRC*)	–	99.58

The null hypothesis H_0 says that the hypervolume indicators of the approximation sets obtained by A_i and A_j have the same distribution. The significance level α of all rejections is 5%. Based on the given results, $\hat{\alpha}$ is the minimum level of significance level at which H_0 would be rejected

Table 4 Comparison of heuristic approach A_8 with exact approach εLBB on eight small instances

|*B*|	|*T*|	I_{HV} *εLBB*	I_{HV} A_8	ΔI_{HV}	$\|\Omega^*\|$	Found by A_8	Time (s) *εLBB*
20	5	0.8576	0.8576	0.0000	7	7	1
	20	0.6095	0.6029	0.0066	11	6	2
40	20	0.8169	0.8126	0.0043	13	6	44
	40	0.5677	0.5639	0.0038	12	3	112
60	20	0.8652	0.8537	0.0115	17	5	2,975
	40	0.6988	0.6913	0.0075	10	0	362
80	20	0.8915	0.8870	0.0045	17	2	19,461
	40	–	–	–	–	–	>86,400

All runs of A_8 were terminated after 5 min (300 s). The runs of εLBB were terminated after 24 h (86,400 s), if the computation of the Pareto set has not been finished by then

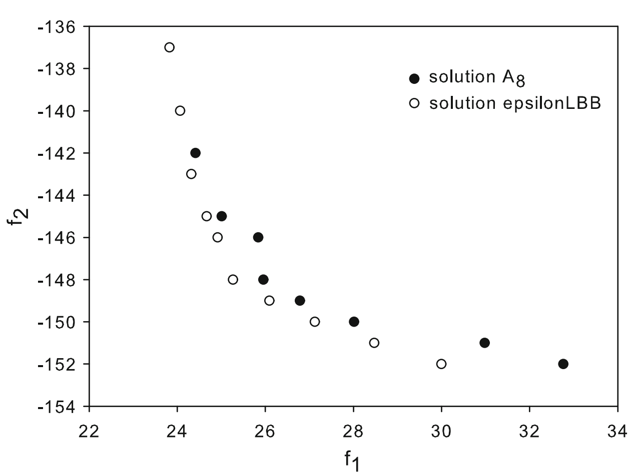

Fig. 3 Comparison of solutions found by A_8 and *εLBB* for the instance with 60 bundle, bids and 40 contracts

Obviously, A_8 is indeed able to find solutions at the same level of f_2 like *εLBB*, but at the cost of higher values of f_1. This effect seems to intensify for decreasing values of f_2. This provides an indication that developing the cost-reducing abilities of the problem-specific core heuristics could further improve the GA's performance.

5 Conclusions and outlook

In this study, a model for a bi-objective winner determination problem in combinatorial transportation procurement auctions was presented. The model, which is based on a set covering formulation, simultaneously minimises total procurement costs and maximises the service-quality level of the execution of all transportation contracts.

To solve this model, two algorithms were introduced. On the one hand, an exact bi-objective branch-and-bound algorithm was proposed following the epsilon constraint approach. On the other hand, the well-known multiobjective evolutionary algorithm SPEA2 was extended by a set of problem-specific evolutionary operators to solve the 2WDP-SC. By differently combining these operators, eight variants of this genetic algorithm were constructed.

The performance of the algorithms was evaluated on a set of newly generated test instances. The test instances were designed to reflect important economic properties of the transportation domain, e.g., free disposal and strict subadditivity of the submitted bids.

The exact branch-and-bound algorithm finds optimal solutions only for small instances in reasonable time and therefore proved unsuitable for transportation procurement auctions of practical dimensions. The relative performance of the eight MOGA variants was evaluated on the large problem instances. The results show a strong dependence of the MOGA performance on the quality of the initial population. Unless the population is initialised using the more elaborated heuristics, even the intelligent operators do not compensate for the losses in solution quality. The best genetic algorithm was also compared to the results of the exact algorithm for the small instances. For these instances, the genetic algorithm was able to generate solutions in or close to the true Pareto solution set.

Our ongoing and future work on this topic takes the following directions. In order to improve the performance of the exact approach, calculation of lower bounds is being enhanced using heuristics. In addition, another exact approach instead of the sequential epsilon-constraint approach is being developed which simultaneously optimises both objectives. As to the heuristic approach, the generic crossover and mutation operators of the GA still leave room for improvement by integrating problem-specific knowledge. This also could mitigate the sensitivity of the GA to the quality of the initial population. Furthermore, an alternative heuristic approach is being developed based on advanced neighbourhood search techniques. Finally, several ways to integrate both exact and heuristic approaches are being intensively explored. For example, overall performance effects caused by seeding the exact approach with bounds derived from the solutions found by different construction heuristics are being studied. On the other hand, using an exact approach to repair infeasible offspring of a GA appears promising.

References

1. Abrache J, Crainic T, Rekik MGM (2007) Combinatorial auctions. Ann Oper Res 153(34):131–164
2. Buer T, Pankratz G (2008) Ein pareto-optimierungsverfahren für ein mehrkriterielles gewinnerermittlungsproblem in einer kombinatorischen transportausschreibung. In: Bortfeldt A, Homberger J, Kopfer H, Pankratz G, Strangmeier R (eds) Intelligente Entscheidungsunterstützung. Gabler Verlag, Wiesbaden, pp 113–135
3. Caplice C, Sheffi Y (2003) Optimization-based procurement for transportation services. J Business Logist 24(2):109–128
4. Caplice C, Sheffi Y (2006) Combinatorial auctions for truckload transportation. In: Cramton P, Shoaham Y, Steinberg R (eds) (2006) MIT Press, Cambridge, pp 539–571
5. Cargoclix Dr. Meier & Schmidt GmbH (2010) Success stories. http://www.cargoclix.com/info/en/success-stories/page.html. Last accessed 22 March 2010
6. Chankong V, Haimes YY (1983) Multiobjective decision making: theory and methodology. Wiley, New York
7. Cramton P, Shoaham Y, Steinberg R (eds) (2006) Combinatorial auctions, MIT Press, Cambridge, MA
8. Ehrgott M, Fonseca CM, Gandibleux X, Hao JK, Sevaux M (eds) (2009) Evolutionary multi-criterion optimization, fifth international conference, EMO 2009, Nantes, April 2009, Proceedings, Lecture notes in computer science, vol 5467. Springer
9. Eiben A, Smith J (2003) Introduction to evolutionary computing. Springer, Berlin
10. Elmaghraby W, Keskinocak P (2004) Combinatorial auctions in procurement. In: Harrison T, Lee H, Neale J (eds) The practice of supply chain management: where theory and application converge. Springer, New York, pp 245–258
11. ETH Zurich (2009) ETH Zurich, System optimization, Pisa. ETH Zurich, System Optimization. http://www.tik.ee.ethz.ch/sop/pisa/. Last accessed 30 January 2009
12. Feo T, Resende M (2005) Greedy randomized adaptive search procedures. J Glob Optim 6:109–133
13. Haimes Y, Lasdon L, Wismer D (1971) On a bicriterion formulation of the problems of integrated system identification and system optimization. IEEE Trans Syst Man Cybern 1(3):296–297
14. Hudson B, Sandholm T (2004) Effectiveness of query types and policies for preference elicitation in combinatorial auctions. In: 3rd international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), IEEE Computer Society, Washington, pp 386–393
15. Ledyard JO, Olson M, Porter D, Swanson JA, Torma DP (2002) The first use of a combined value auction for transportation services. Interfaces 32:4–12
16. Leyton-Brown K, Shoham Y (2006) A test suite for combinatorial auctions. In: Cramton P, Shoaham Y, Steinberg R (eds) Combinatorial auctions, MIT Press, Cambridge, pp 451–478
17. Meisell MJ, Norbis M (2008) A review of the transportation mode choice and carrier selection literature. Int J Logist Manag 19(2):183–2111
18. Nisan N (2000) Bidding and allocation in combinatorial auctions. In: EC '00: Proceedings of the 2nd ACM conference on Electronic commerce, pp 1–12
19. Sandholm T, Suri S, Gilpin A, Levine D (2002) Winner determination in combinatorial auctions generalizations. In: International conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Bologna, pp 69–76
20. Sheffi Y (2004) Combinatorial auctions in the procurement of transportation services. Interfaces 34(4):245–252
21. Song J, Regan A (2004) Combinatorial auctions for transportation service procurement: the carrier perspective. Transp Res Record 1833:40–46
22. Suhl L, Mellouli T (2009) Optimierungssysteme—Modelle, Verfahren, Software, Anwendungen. Springer, Berlin
23. Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 3(4):257–271
24. Zitzler E, Laumanns M, Thiele L (2002) Spea2: improving the strength pareto evolutionary algorithm for multiobjective optimization. In: Giannakoglou K, Tsahalis D, Periaux J, Papailiou K, Fogarty T (eds) Proceedings of the EUROGEN2001 conference, CIMNE, Barcelona, pp 95–100
25. Zitzler E, Thiele L, Laumanns M, Fonseca CM, da Fonseca VG (2003) Performance assessment of multiobjective optimizers: An analysis and review. IEEE Trans Evol Comput 7:117–132